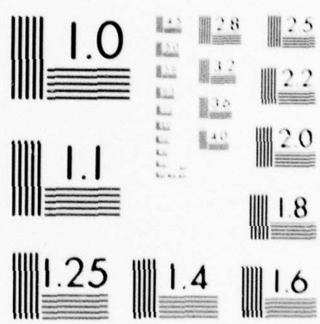
AD-A072 565 ALABAMA UNIV UNIVERSITY BUREAU OF ENGINEERING RESEARCH F/G 15/5
DISCRETE OPTIMIZATION WITH NONSEPARABLE FUNCTIONS; DETERMINATIO--ETC(U)
JUL 79 D CHEN AFOSR-78-3549
UNCLASSIFIED BER-237-69 AFOSR-TR-79-0917 NL

| OF |

AD
A072565

END
DATE
FILMED
9-79
DDC


1.0
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

AD A072565

DDC FILE COPY

19 REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: 18 AFOSR-TR-79-0917	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): 6 DISCRETE OPTIMIZATION WITH NONSEPARABLE FUNCTIONS; DETERMINATION OF A KIT COMPOSITION FOR WAR READINESS SPARE PARTS.		5. TYPE OF REPORT & PERIOD COVERED: 9 FINAL rept.
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): 10 Der-San/Chen		8. CONTRACT OR GRANT NUMBER(s): 15 AFOSR-78-3549
9. PERFORMING ORGANIZATION NAME AND ADDRESS: The University of Alabama BER College of Engineering University, Alabama 35486		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 16 17 61102F 2304/A6
11. CONTROLLING OFFICE NAME AND ADDRESS: Air Force Office of Scientific Research/NM Bolling AFB, Washington, D.C. 20332		12. REPORT DATE: 11 July 1979
		13. NUMBER OF PAGES: 54 12 61p.
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): 14 BER-237-69		15. SECURITY CLASS. (of this report): UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

War Readiness Spares Kit, Discrete Optimization, Nonlinear Constraints, Separable and Nonseparable Functions, Branch-and-Bound Techniques, Local and Global Optimum

067500 JOB

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This research is intended to improve the author's previous algorithm for solving an optimization problem connected with the determination of the composition of a War Readiness Spares Kit. The problem was formulated as a discrete minimization model with two nonlinear constraint functions, of which one is nonseparable.

The solution algorithm is modified in several aspects including selection of a good initial kit composition, improvements in convergence toward a global optimum and refinements in programming techniques. A branch-and-bound (cont)

20. Abstract (cont)

technique and a univariate search method are incorporated into the algorithm and a numerical example is given.

Computational results show that the revised algorithm is improved in computational efficiency. However, some factors must be considered in order to enhance the capability of the algorithm for solving practical problems.

FINAL REPORT

on

Contract AFOSR-78-3549

DISCRETE OPTIMIZATION WITH NONSEPARABLE FUNCTIONS;

DETERMINATION OF A KIT COMPOSITION FOR WAR READINESS SPARE PARTS

by

Der-San Chen, Associate Professor
Principal Investigator
Department of Industrial Engineering
The University of Alabama

Prepared for

United States Air Force
Air Force Office of Scientific Research
Bolling AFB, D.C. 20332

July 1979

BER Report No. 237-69

DISCLAIMER NOTICE

FINAL REPORT

on

Contract AFOSR-78-3549

DISCRETE OPTIMIZATION WITH NONSEPARABLE FUNCTIONS;

DETERMINATION OF A KIT COMPOSITION FOR WAR READINESS SPARE PARTS

by

Der-San Chen, Associate Professor
Principal Investigator
Department of Industrial Engineering
The University of Alabama

Prepared for

United States Air Force
Air Force Office of Scientific Research
Bolling AFB, D.C. 20332

July 1979

BER Report No. 237-69

ABSTRACT

This research is intended to improve the author's previous algorithm for solving an optimization problem connected with the determination of the composition of a War Readiness Spares Kit. The problem was formulated as a discrete minimization model with two nonlinear constraint functions, of which one is nonseparable.

The solution algorithm is modified in several aspects including selection of a good initial kit composition, improvements in convergence toward a global optimum and refinements in programming techniques. A branch-and-bound technique and a univariate search method are incorporated into the algorithm and a numerical example is given.

Computational results show that the revised algorithm is improved in computational efficiency. However, some factors must be considered in order to enhance the capability of the algorithm for solving practical problems.

CONTENTS

I. INTRODUCTION

This research project is a sequel to the author's previous work [1] that developed an algorithm for solving an optimization problem connected with the determination of the composition of a War Readiness Spares Kit (WRSK). The problem was formulated as a cost minimization model with non-linear constraints in which the decision variables are required to be integers.

The model and its variants are rather appealing to the optimization theoreticians as well as to the logistics practitioners. Theoretically, they belong to a class of discrete optimization problems which at present have no efficient solution algorithms. In application, this model can represent many important concrete problems in the logistics area [2,3].

The purpose of this research is to improve the efficiency of the rudimentary algorithm which was previously proposed by the author [1].

The problem under study will be briefly described and its associated optimization model will be derived in the next two sections. Section IV will describe other approaches to this problem and will review the previously proposed solution algorithm to our optimization model [1]. An improved version of the algorithm will be presented and illustrated by a numerical example in Section V. Finally, computational results and conclusions will be reported in Section VI, and a listing of FORTRAN programs for the algorithm included as an appendix.

II. PROBLEM STATEMENT

A War Readiness Spares Kit for an aircraft squadron consists of selected spare parts required to sustain its wartime activities during the period when the squadron is operating under a "remove and replace" maintenance concept. Any items that fail in the aircraft are removed from it and replaced with spare items from the kit or with serviceable items from another "down" aircraft. These down aircraft are referred to as NORS (Not Operationally Ready, Supply) aircraft. The problem is to determine the quantity for each selected item to be placed in the kit so that the total cost per kit is minimized while maintaining a desirable level of unit readiness.

Two criteria are currently in use for measuring unit readiness: (1) the expected (or average) number of NORS aircraft and (2) the total of the expected shortages for all items. The expected NORS seems to be more logical [4] in terms of measuring the readiness of a squadron, whereas the total expected number of shortages is easier in computation and is a traditional method used by Air Force supply personnel. Both measures of readiness are concurrently used in practice. However, confusion may arise whenever the performance of a kit is satisfactory in one measure and unsatisfactory in the other. As a result, a combined measure has been proposed [1]. The optimization model derived in this study is based on this measure.

III. AN OPTIMIZATION MODEL

The WRSK problem can be formulated as the following optimization model: Given certain acceptable levels of expected number of NORS aircraft (b_1) and the total expected shortages for all items (b_2), find a kit composition, $X = (x_1, x_2, --, x_i, --, x_r)$ such that the total cost is minimized. Mathematically, the model is one of a class of discrete resource allocation problems;

$$\begin{aligned} \text{Minimize} \quad & Z(X) = \sum_{i=1}^{r} c_i x_i \\ \text{Subject to} \quad & E_1(X) \leq b_1 \\ & E_2(X) \leq b_2 \\ & x_i \geq 0 \text{ and integer} \end{aligned} \tag{1}$$

where:

x_i = quantity of a line item, i, to be placed in the kit

c_i = unit cost of line item i

$E_1(X)$ = computed value of the expected number of NORS aircraft at composition level X

$E_2(X)$ = computed value of the total expected shortages at composition level X

The model is an n-dimensional discrete optimization problem with two nonlinear constraints. Logically, the constraint functions must be decreased as the number of units in the kit increase. The decreasing property will become more evident when the functions of $E_1(X)$ and $E_2(X)$ are algebraically defined.

Assumptions

The following assumptions are made in the derivation of $E_1(X)$ and $E_2(X)$:

1. The number of failures for each line item within any time interval is a random variable which follows a Poisson distribution.
2. When a failure (or demand) of a part cannot be filled from the kit, then it is satisfied by removing the needed part, if available, from an already NORS aircraft. Thus, parts shortages are consolidated on as few aircraft as possible.
3. All down aircraft have approximately the same number of serviceable items.
4. Every item placed in the kit is essential to the operation of an aircraft. Failure of any such item could cause a NORS aircraft.
5. All items are independent in the sense that the failure of any item does not affect the failure of any other item.

Expected Number of NORS Aircraft

Under assumption 1, the number of failures which occur in a unit time follows the Poisson distribution with mean λ. If $q(j)$ is the probability of exactly j failures in a unit time, then:

$$q_j = \frac{\lambda^j e^{-\lambda}}{j!}$$

and the probability of i or fewer failures is:

$$Q(i) = \sum_{j=0}^{i} q_j$$

Let n be the number of down aircraft for lack of essential parts and a_i be the quantity of item i available in each aircraft. If item i has x_i spares in the kit, then under assumptions 2 and 3, there are a total of $(x_i + na_i)$ spares available for replacement of item i.

Since each item is essential to the operation of aircraft (assumption 4) and all items are independent (assumption 5), the probability of n or less NORS aircraft is:

$$\prod_{i=1}^{r} Q_i(x_i + na_i) \quad \text{where } r = \text{number of item types}$$

and the probability of exactly n NORS aircraft is:

$$\prod_{i=1}^{r} Q_i(x_i + na_i) - \prod_{i=1}^{r} Q_i(x_i + (n-1)a_i)$$

Then by definition, the expected number of NORS aircraft in a squadron of N aircraft is:

$$E_1(X) = \sum_{n=0}^{N} n \left[\prod_{i=1}^{r} Q_i(x_i + na_i) - \prod_{i=1}^{r} Q_i(x_i + (n-1)a_i)\right] \tag{2}$$

Note that:

$$\prod_{i=1}^{r} Q_i(x_i + N\, a_i) = 1$$

Expanding and rearranging (2), we obtain the following expression:

$$E_1(X) = N - \sum_{n=0}^{N-1} \prod_{i=1}^{r} Q_i(x_i + na_i) \tag{3}$$

where:

$$Q_i(x_i + na_i) = \sum_{j=1}^{x_i+na_i} q_j$$

A function F of several variables, $x_1, x_2, \ldots, x_r$ is called separable if there are n functions, $f_1, f_2, \ldots, f_r$ of one variable each, such that

$$F(x_1, x_2, \ldots, x_r) = \sum_{i=1}^{r} f_i(x_i)$$

A very pleasant property of separable functions is that they may be optimized one variable at a time. This property can lead to considerable savings in computational time; in fact, it can make the difference between

a possible and an impossible computation. For example, if $r = 100$ and each of the variables $x_1, x_2, \ldots, x_r$ can take on 10 different values, then we need only look at 10 values of x_1 in order to optimize f_1, 10 values of x_2 in order to optimize f_2, and so on, leading in total of 1,000 different values of variables $x_1, x_2, \ldots, x_r$. This is a possible task. If F is not separable, however, we must look at all of the 10^{100} different values. This is an impossible task even on a modern high speed computer.

Unfortunately, the problem we have here is nonseparable. Note that the number of aircraft (N) in a squadron is a constant. The second term of $E_1(X)$ is a nonseparable function. This nonseparability causes the difficulty in computation because the state of the art of the solution methods to this problem is still primitive.

Another property pertaining to the function $E_1(X)$ is that it is monotonic decreasing. This can be verified easily. Consider the difference in functional value when x_i $(i=1, 2, \ldots, r)$ increases by one unit,

$$E_1(x_1,x_2,\ldots,x_i+1,\ldots,x_r) - E_1(x_1,x_2,\ldots,x_i,\ldots,x_r)$$

$$= - \sum_{n=0}^{N-1} q_{x_i+na_i+1} \quad Q_i(x_i+na_i) \tag{4}$$

Since q and Q are respectively Poisson probability density function and cumulative probability function, they must be strictly positive unless $q_\infty=0$ or $Q(\cdot)=q_\infty$. In our problem, we deal with a finite number of n and a_i which implies that both q and Q are strictly positive. Therefore equation (4) is strictly negative, and the function is monotonic decreasing.

Total Expected Number of Shortages

Now let us define the total expected shortages for all items. A shortage occurs whenever the number of demands (or failures) exceeds the

number of spares in the kit. The total number of demands for a squadron of N aircraft cannot exceed $(Na_i + k_i)$, denoted by j_{max}. Then by definition the expected number of shortages for item i alone is:

$$f_i = \sum_{j_i=x_i+1}^{j_{max}} (j_i - x_i)\, q_{j_i} + (j_{max} - x_i)\left[1 - \sum_{j_i=0}^{j_{max}} q_{j_i}\right] \qquad (5)$$

The last term of (5) is a correction factor for the tail of the Poisson distribution.*

Then the total expected number of shortages for all items can be obtained by summing (5) over i, i.e.,

$$E_2(X) = \sum_{i=1}^{r} f_i \qquad (6)$$

Fortunately, the function of $E_2(X)$ is separable. It is also a monotonic decreasing function, since

$$E_2(X+e_i) - E_2(X)$$

$$= - \sum_{j=x_i+1}^{j_{max}} q_j < 0 \qquad (7)$$

where e_i = the unit vector with 1 in element i.

---

*An alternative method of treating the truncated tail is to normalize the probabilities by dividing the first term by:

$$\sum_{j_i=0}^{j_{max}} q_{j_i} \quad .$$

Since the selection of the correction methods is not the primary concern of this study and since (5) is currently being used in D029 of the AF Logistics Command, we shall use that definition henceforth.

IV. EXISTING APPROACHES AND SOLUTION ALGORITHMS

In this section, two approaches currently in use for the determination of the kit composition are discussed. Additionally, a previously proposed solution algorithm to our optimization model is presented.

Conventional Method [5]

Traditionally, the kit is composed of the essential items in the amount of their respective mean failure rates per x number of flying hours. This is rounded to the nearest integer number provided that every item type has a minimum of one unit per kit.

The advantage of this method is its simplicity in computation. However, the kit composition thus determined may be far from the optimum because it completely ignores the requirements of meeting acceptable levels of readiness and budgetary limitations in allocation of the spare parts.

Marginal Analysis [5]

The method of marginal analysis (or incremental analysis) is an iterative procedure consisting of the following basic steps:

1. Begin with the empty kit composition, $X^{\circ} = (0,0,\ldots,0)$, and set $v(X^{\circ}) = \infty$.
2. At iteration t, compute the change in performance per unit cost for all i,

$$\Delta_i = \frac{v(X^{t-1}) - v(X^{t-1} + e_i)}{c_i} \tag{8}$$

where

$$v(X^S) = \alpha \cdot E_1(X^S) + E_2(X^S) \tag{9}$$

3. Find $j = \{i | \max_{i=1,2,\ldots,r} \Delta_i\}$, and compute:

$$X^t = X^{t-1} + e_j$$

$$Z(X^t) = Z(X^{t-1}) + c_j$$

4. Increase t by 1 and repeat steps 2 and 3 until max Δ_i is equal to 0, or exceeds the prescribed allowance.

Conversely, we may begin with a large initial kit composition and decrease one unit at a time in the direction of minimizing Δ_i, until no reduction is possible. Existing kit compositions may also serve as a basis for incremental adjustment. When a large initial kit surpasses the acceptable readiness level, the decrement version of the method is used; when the kit composition falls short, the increment version is utilized.

The basic method of marginal analysis and its variations consider the cost of a kit as well as the improvement of the readiness by both measures. However, different initial kits will result in different final kit compositions, which, in turn, vary greatly in terms of cost and level of readiness.

Another problem of the method is in the selection of an appropriate weight (α) for formula (9) since it also affects the final solution. Empirical computer tests [1] of different combinations of the initial kits and α-weightings indicated that the differences in final kit compositions were very significant.

Original Algorithm

An algorithm was proposed by the author in his previous work [1] in an attempt to find an optimum kit composition. The algorithm utilizes the computational efficiency of marginal analysis to determine a good, feasible kit by trying various combinations of initial kit compositions and

weights. The final solution thus obtained is referred to as a local or relative optimum (denoted by X°) since X° yields total cost $Z(X^\circ) \leq Z(X^\circ \pm e_i)$ for all i while subject to $E_1(X^\circ) \leq b_1$ and $E_2(X^\circ) \leq b_2$.

The implication is that no reduction in cost is possible if the search is continued in the univariate direction.

In an attempt to further improve the solution, it was proposed that a hyperplane passing through X° be constructed and searched for another feasible integer point. Finding these points turns out to be a knapsack problem [1]. If a feasible integer point is found, then the method of marginal analysis is again applied to obtain a new local optimum, which is at least as good as the previous one. If no feasible integer point can be found, the hyperplane is moved in a parallel manner by reduction of cost. These two steps are alternately repeated until the amount of the cost reduction is equal to or less than max c_i. The final feasible solution is a global (or absolute) optimum.

Knapsack Problem

Mathematically, finding the X integer points on a hyperplane is equivalent to finding $x_i = 0,1,2,..$ which solves

$$\sum_r c_i x_i = Z^* \qquad (10)$$

where Z^* is the current local minimum cost found by the method of marginal analysis. In order to apply the existing algorithms for this knapsack problem, an objective function is added: Min $\Sigma\, c_i x_i$. The algorithmic method used is that of the dynamic programming approach [6, 7, 8, 9, 10].

Absolute Lower Bounds

In order to reduce the problem size of (10), a procedure was used to establish an absolute lower bound for each of the item types.

Let L_i be a lower bound for item i. Substituting $x_i' = x_i - L_i$ into (10), we obtain the following transformed equation,

$$\sum_{i=1}^{r} c_i x_i' = Z^* - \sum_{i}^{r} c_i L_i \tag{11}$$

Note that the righthand side of the equation (11) is reduced, as is the problem size.

We shall now show how the absolute lower bound , L_i, may be derived from $E_1(X) \leq b_1$ and $E_2(X) \leq b_2$.

Consider the constraint of the expected number of NORS aircraft,

$$E_1(X) = N - \sum_{n}^{N-1} \prod_{i}^{r} Q_i\,(x_i + na_i) \leq b_1 \tag{12}$$

To obtain an absolute lower bound for item p, we let all other items be sufficiently large such that $Q_i(x_i + na_i) = 1$ as $x_i = \infty$ in which case, we obtain a reduced constraint involving a single variable x_p:

$$N - \sum_{n=0}^{N-1} Q_p\,(x_p + na_p) \leq b_1 \tag{13}$$

Note that the left-hand-side function is monotonically decreasing in x_p. Finding the lower bound L_p' is equivalent to finding the smallest value of p that satisfies (13). Mathematically,

$$L_p' = \text{Min}\,\{x_p | N - \sum_{n=0}^{N-1} Q_p(x_p + na_p) \leq b_1\} \tag{14}$$

To find L_p' computationally, we may begin with $x_p = 0$ and increase x_p by one unit until the constraint is satisfied. More efficiently, the interval bisection method may be applied when the value of L_p' is expected to be large.

Another absolute lower bound may be similarly derived from the constraint of total expected shortages in (5), i.e.

$$\sum_{j_p = x_p+1}^{x_p + Na_p} (j_p - x_p)\, q_p + Na_p[1 - Q_p(x_p + Na_p)] \leq b_2$$

Since the above function is monotonically decreasing in x_p, the lower bound L_p'' may be calculated by finding the smallest value of p, i.e.

$$L_p'' = \text{Min}\ \{x_p \mid \sum_{j_p = x_p+1}^{x_p + Na_p} (j_p - x_p)\, q_p - Na_p \cdot Q(x_p + Na_p) \leq b_2 - Na_p\} \tag{15}$$

To obtain a higher lower bound for item p (p = 1,2,...,r), the larger of L_p' and L_p'' is selected.

<u>Summary</u>

The previous algorithm may be summarized by a flowchart shown in Figure 1. The input datum include the number of aircraft in a squadron (N), the unit cost (c_i), the number of units per item type per aircraft (a_i) and the mean failure rate (λ_i) for all item i = 1,2,...,r.

Let b_1 and b_2 respectively be the computed values of the expected NORS aircraft and the total expected shortages of the composition, determined by the conventional method. These values will be treated as the acceptable levels defined in (1). Attempts are made to find a kit composition that maintains at least the same levels of b_1 and b_2 yet requires a minimum cost investment.

First, an absolute lower bound for each item i = 1,2,...,r is established by equations (14) and (15).

Then, a kit composition is determined via the method of marginal analysis. In order to insure a good final kit composition, many different combinations of initial kit composition and various relative weights (α) are tested, and the best kit composition is selected.

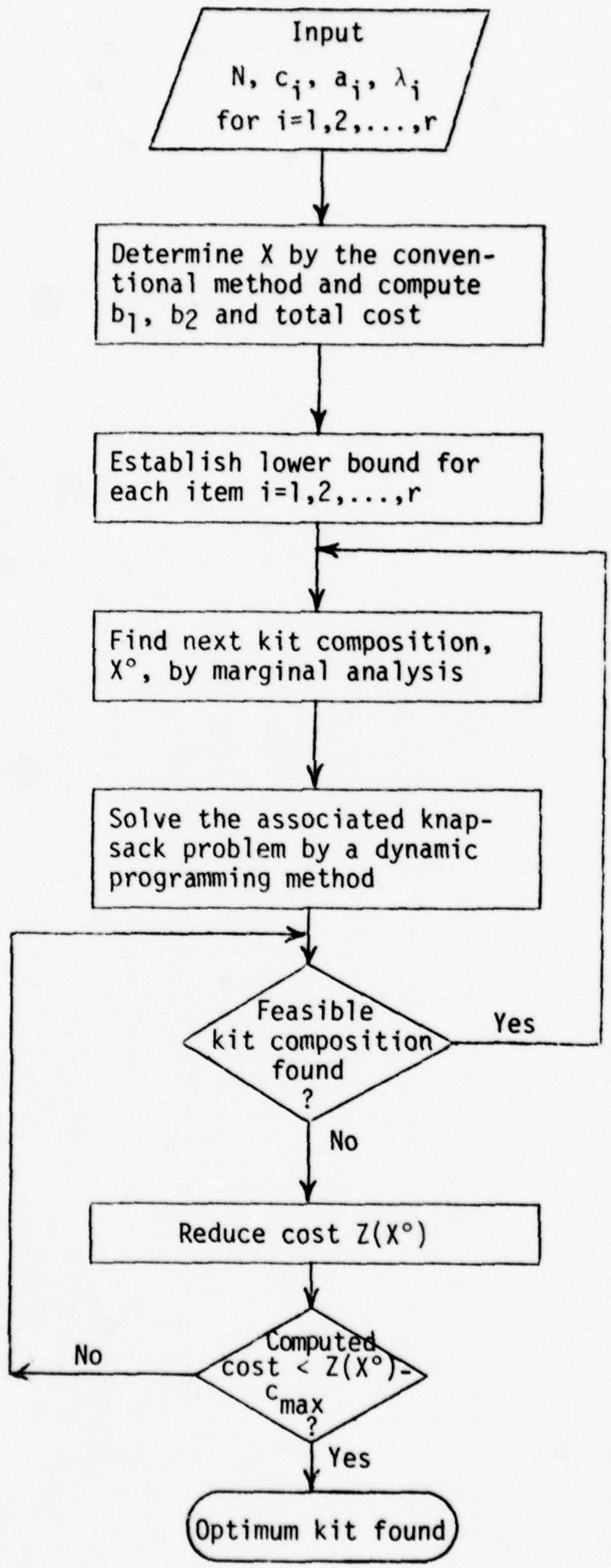


Figure 1. Flowchart for the Original Algorithm.

A knapsack problem is then constructed to generate another feasible integer solution or point on the same hyperplane (i.e., with the same cost). If such a solution is located, the method of marginal analysis is again tried to reduce the cost. When no feasible solutions can be found on the current hyperplane, the search for a feasible point is continued on a less-cost hyperplane. The procedure is repeated until the total cost reduction exceeds the maximum unit cost for all items, denoted by max c_i. Any search for a feasible solution below the cost, Z°-max c_i, is unnecessary because of the strictly decreasing property of the constraint functions and the strictly increasing property of the cost function.

Previous Computational Results

The algorithm was programmed and tested on Honeywell 6000/600 computer at Wright-Patterson AFB, and later on UNIVAC-1110 at The University of Alabama. The computational results showed that the requirements of computer memory and computation time increased exponentially as the number of item types increased. For instance, the computation time for solving a kit composed of 10 different item types took about 20 minutes in UNIVAC-1110. A problem of 11 different items, required about 45 minutes. For a larger problem size, the optimum solution cannot be found in a reasonable amount of computer time.

V. REVISED ALGORITHM

In this section, the weakness of the original algorithm will be identified and improved. The improvements include selection of a good initial kit composition, improvements in convergence toward a global optimum, and refinements in programming techniques.

Initial Kit Composition

In order to obtain a "good" starting kit composition, the original algorithm applied the method of marginal analysis to different combinations of the initial kit composition and weighting. This procedure not only caused computational ineffectiveness but also obtained an unpredictable result. The revised algorithm utilizes the kit composition determined by the conventional method to serve as a starting point. In fact, empirical tests show that this kit composition, on many occasions, is better than local minima found by marginal analysis.

Univariate Search

In an attempt to find a local minimum, the original algorithm steers the search at each iteration in a univariate direction (via marginal analysis) that maximizes the incremental gain in the combined performance per unit cost. Although these moves can finally lead to a feasible solution, it may however be far from the optimum.

Instead of using marginal analysis, the revised algorithm reduces the cost within the feasible region by moving in a univariate direction that yields the greatest unit reduction. Let X^k and X^{k+1} be the feasible kit composition in two successive iterations k and k+1. Their relationship is:

$$X^{k+1} = X^k - e_j$$

where:

$$j = \{i \mid \max c_i,\ E_1(X^{k+1}) \leq b_1 \text{ and } E_2(X^{k+1}) \leq b_2\} \tag{16}$$

Search Region

The greatest weakness of the original algorithm is in the searching for a feasible integer point after a local optimum has been found. If no feasible point is found, the searching must be continued on a family of parallel hyperplanes. Since the number of integer points on these hyperplanes may be astronomical, examination of them is computationally intractable, if not impossible, even if the dynamic programming approach is utilized.

In order to circumvent this problem, two remedial actions are taken: (1) narrow down the cost range to be searched from max c_i to min c_i; and (2) restrict the integer points to be searched to those delimited by the upper and lower bounds for each x_i and the constraints of (6) and (12). Hopefully, a large subset of infeasible points (or ungainful points) will be eliminated from examination. In what follows, we shall validate that the search is sufficient within the cost range of min c_i.

Theorem: Let $E_1(X)$ and $E_2(X)$ be monotonic decreasing functions and $Z(X) = CX$ a monotonic increasing function in X, where $C = (c_1, c_2, \ldots, c_i, \ldots c_r)$ and $c_i > 0$ for all i. Also let X_1 be a feasible solution to problem (1), i.e.,

$$X_1 \varepsilon S = \{X \mid (E_1(X) \leq b_1) \cap (E_2(X) \leq b_2)\}$$

with cost $Z(X_1) = CX_1$. If there exists $X_3 \varepsilon S$ with $Z(X_3) = CX_3 < Z(X_1)$, then at least one $X \varepsilon S$, say X_2, can be found in the interval

$$[Z(X_1) - c_k,\ Z(X_1)),$$

where

$$c_k = \min c_i.$$

Proof: There are but two cases of $X_3 \varepsilon S$:

(1) $Z(X_1) - c_k \leq Z(X_3) < Z(X_1)$ and

(2) $Z(X_3) < Z(X_1) - c_k < Z(X_1)$

Obviously, the theorem holds for case (1) if we let $X_3 = X_2$. For case (2), we prove the theorem by contradiction.

Assume $X_3 \varepsilon S$ with $Z(X_3) < Z(X_1)-c_k$ and there exists no feasible solution in the interval, $[Z(X_1) - c_k, Z(X_1))$. Let e_k be a unit vector associate with c_k. Then we can always find an integer $m = 1, 2, \ldots,$ such that $(X_3 + me_k) \varepsilon S$ with $Z(X_3 + me_k) < Z(X_1)$ and $Z(X_3 + (m+1)e_k) > Z(X_1)$ since $E_1(X)$ and $E_2(X)$ are monotonic decreasing functions and $Z(X)$ is a monotonic increasing function.

Since $Z(X_3 + (m+1)e_k) = Z(X_3 + me_k) + c_k$, it follows $Z(X_3 + me_k) > Z(X_1) - c_k$. By letting $X_2 = X_3 + me_k$, we have a feasible solution $X_2 \varepsilon S$ found in an open interval $(Z(X_1) - c_k, Z(X_1))$, a subinterval of $[Z(X_1) - c_k, Z(X_1))$, which contradicts our assumption. Q.E.D.

One application of this theorem is that if we cannot find a feasible solution in the interval $[Z(X_1) - \min c_i, Z(X_1))$, then no better feasible solution can be found and therefore the current local optimum is also a global optimum.

Branch-And-Bound Technique

We shall now develop a systematic scheme that can search for a feasible integer solution, if one exists, in the interval of $[Z(X_1) - c_k, Z(X_1))$. An integer point is feasible if it lies within the region delimited by this interval, and is subject to the constraints of $E_1(X) \leq b_1$ and $E_2(X) \leq b_2$ in addition to the upper and lower bounds for all variables.

A lower bound (L_p) for x_p can be determined by expression (14) and (15). A upper bound (U_p) for x_p may be derived from the inequality: $\Sigma c_i x_i \leq Z(X_1)$, i.e.

$$U_p = [\{Z(X_1) - \sum_{i=p} L_i\ x_i\}/c_p] \tag{17}$$

where [w] = greatest integer less than or equal to w.

Another upper bound for x_p also may be derived from the property of the Poisson distribution embedded in $E_1(X)$ and $E_2(X)$. This derivation will be given in a later section.

The decision variable x_i can take on the following integer values: L_i, L_i+1, L_i+2, ..., U_i. If we let $y_i = x_i - L_i$, then y_i can take on only the values: $0,1,2,\ldots,(U_i-L_i)$. For ease of computation, from now on, we shall deal with the variable y_i instead of x_i.

Mathematically, we must find a $Y = (y_1,y_2,\ldots,y_i,\ldots,y_r)$ that satisfies the following constraints:

$$\begin{aligned} &Z_1 \leq \Sigma\ c_i y_i \leq Z_0 \\ &0 \leq y_i \leq U_i - L_i \text{ for all } i \\ &E_j(Y+L) \leq b_j \quad j=1,2 \end{aligned} \tag{18}$$

where

$$L = (L_1,L_2,\ldots,L_i,\ldots,L_r)$$

$$Z_0 = Z(X_1) - \Sigma\ c_i L_i$$

$$Z_1 = Z_0 - \min c_i$$

The feasible solution space of the above problem can be represented by a tree where the starting node is associated with all variables set equal to 0. From this node, a branch is generated for each value of y_1 in the interval of $[0, U_1 - L_1]$. Then, from each of these nodes, all integer values in $[0, U_i - L_i]$ of y_2 branch out. The same procedure is repeated for y_3, y_4, ..., y_r.

Numerical Example

Consider the following problem:

$$
\begin{aligned}
&1569 \le 493y_1 + 873y_2 + 490y_3 + 103y_4 \le 1672 && \text{-- (A)} \\
&E_2(Y + L) \le b_2 && \text{-- (B)} \\
&E_1(Y + L) \le b_1 && \text{-- (C)} \\
&y_1 = 0 \text{ or } 1 && \text{-- (D)} \\
&y_2 = 0 \text{ or } 1 && \text{-- (E)} \\
&y_3 = 0,\ 1 \text{ or } 2 && \text{-- (F)} \\
&y_4 = 0,1,2 \text{ or } 3 && \text{-- (G)}
\end{aligned}
\qquad (19)
$$

A solution tree representing (19D) through (19G) is given in Figure 2. This tree will be used later for describing our search strategy.

A path from the origin node to a terminal node represents an integer solution which may be feasible or infeasible. The tree includes all possible integer solutions. Since this total enumeration is computationally intractable even for a small number of variables, we shall present a branch-and-bound technique that will systematically examine only a small subset of the tree.

We shall use the upper and lower bounds in (19A) to be bounds on a node and use the constraint of expected shortage in (19B) to be a criterion for determining the order of branching. Any subtrees that may branch from this node will be dropped from consideration. Because the computational efforts involved in constraint (19C) are considerable, it will not serve as a bounding constraint and will be checked for feasibility only after a complete solution is obtained.

The first bounding constraint is the upper bound for the current cost, $Z_0 = 1672$. Any subtree which has a minimum cost, $T_1(t)$, greater than Z_0 will not be generated. The value of $T_1(t)$ for node t is defined as follows.

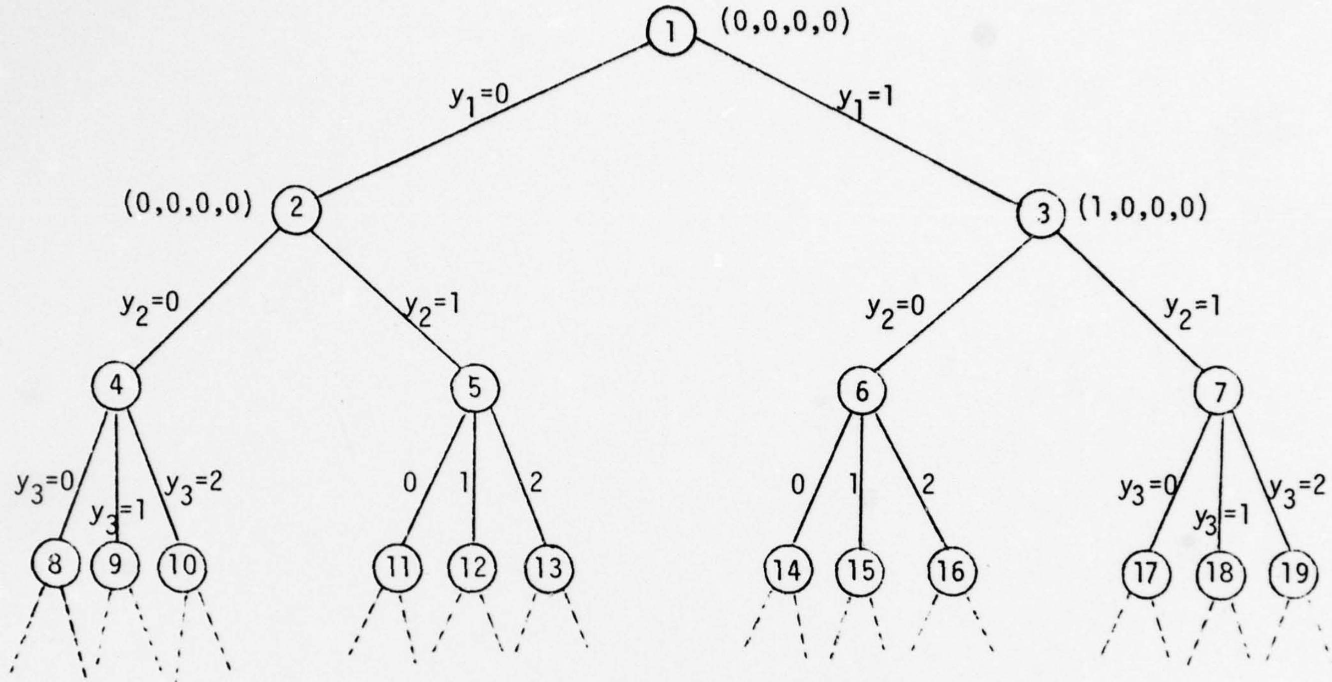


Figure 2. A Complete Solution Tree.

At node t, some variables are specified and some variables are unspecified or free. At node 7, for example, the specified variables are y_1 and y_2 and the free variables are y_3 and y_4. The minimum cost for the subtree branching from node 7 is obtained by setting the free variables equal to 0 because the total cost is always nondecreasing as a variable is added. This minimum cost is denoted by $T_1(t)$ for node t. For example, consider Figure 3 with $T_1(t)$ on nodes. Nodes 18 and 19 can be eliminated because they exceed Z_0 = 1672.

The second bounding constraint is the lower bound for the current cost, Z_1 = 1569. Any subtree branching from node t which has a maximum cost, denoted by $T_2(t)$, less than Z_1 will be cut off. The $T_2(t)$ is defined by setting the free variables equal to their upper bounds (U_i). Figure 4 shows the subtrees to be cut off due to this bound.

We shall now discuss how to determine the node to be generated first. In tree expansion, we start out with node 1 and expand it to generate its successors, nodes 2 and 3. The problem is: which node should be considered such that a solution in the subtree generated from that node will be more likely to be feasible. In order to attain this, an evaluation function [11] defined by $E_2(Y'' + L)$ will be used to estimate the best expected shortages for a given node or subtree. In other words, the evaluation function is used to provide a means for ranking those nodes that are candidates for expansion to determine which one is most likely to be the best path to a feasible solution. The nodes with the smallest functional value will be branched first because the lower the value, the higher the potential to attain a feasible solution. Once a node has been selected, the founding costs associated with the node are computed and the node is checked for the possibility of elimination.

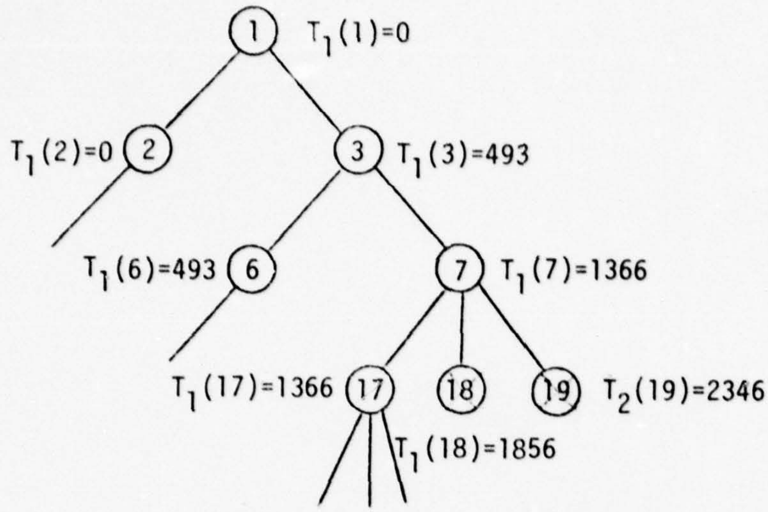


Figure 3. Subtrees Eliminated by Upper Bound Z_0.

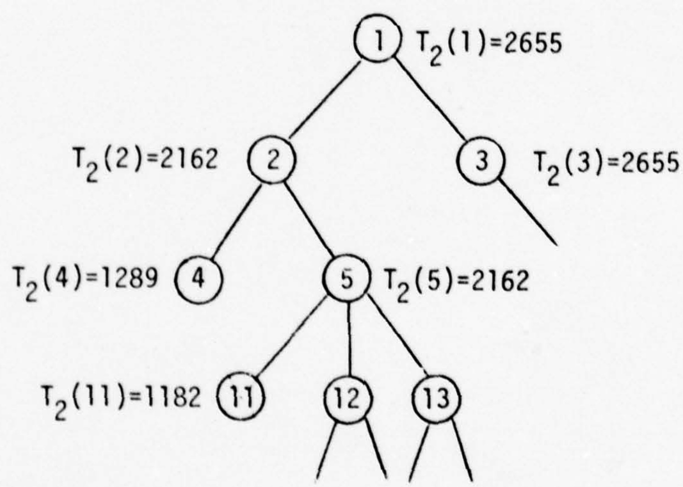


Figure 4. Subtrees Eliminated by Lower Bound Z_1.

In a step-by-step fashion, with the aid of Figure 5, we shall solve problem (19) in which $b_1 = 1.715$, $b_2 = 1.016$ and objective function:

$$\text{minimize } Z = 493y_1 + 873y_2 + 490y_3 + 103y_4.$$

<u>Step 1</u>: Set up the root, A, and compute associated bounding costs and E_2.

$Y'(A) = (0,0,0,0)$, $T_1(A) = 0$ (lower bound for A)

$Y''(A) = (1,1,2,3)$, $T_2(A) = 2655$ (upper bound for A)

$E_2 = (1,1,2,3) = 0$

<u>Step 2</u>: Select the node with min $E_2(Y'')$ and expand this node. Doing so, we obtain nodes B and C:

$Y'(B) = (0,0,0,0)$, $T_1(B) = 0$

$Y''(B) = (0,1,2,3)$, $T_2(B) = 2162$

$E_2(0,1,2,3) = 0.17$

$Y'(C) = (1,0,0,0)$, $T_1(C) = 493$

$Y''(C) = (1,1,2,3)$, $T_2(C) = 2655$

$E_2(1,1,2,3) = 0$

<u>Step 3</u>: Check for a cut-off node. No cut-off node so far.

<u>Step 4</u>: Repeat steps 2 and 3 to generate nodes D through I until a complete solution is obtained. Nodes D, G and H are eliminated as cut-off nodes. Note that at the last level, y_4 is determined by the expression:

$$y_4 = \left[\frac{Z_0 - T_1(F)}{c_4} \right]$$

Since $y_4 = 0$, only one successor is expanded at the last level.

<u>Step 5</u>: Check the complete solution (1,1,0,0) against the constraint of E_1 for feasibility.

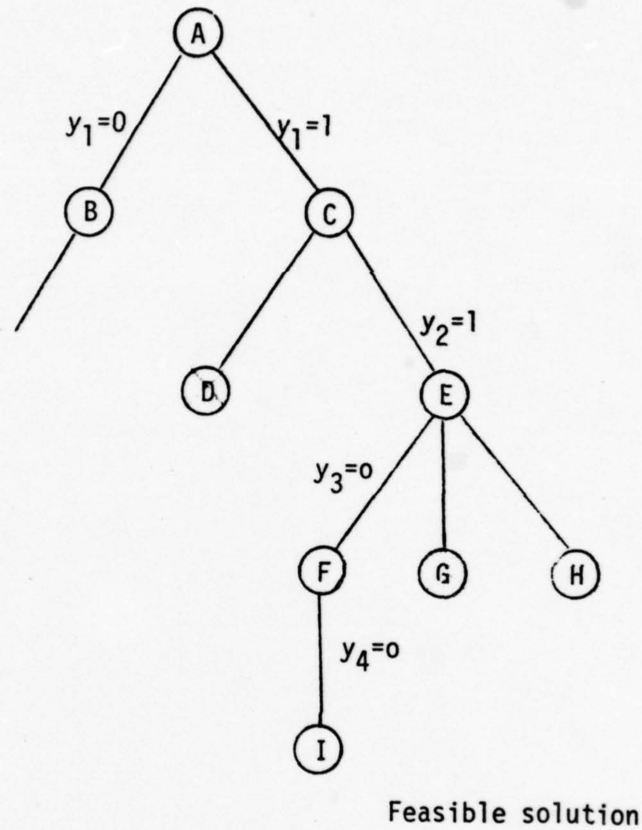


Figure 5. Tree Search for a Feasible Solution.

Should $E_1(1,1,0,0) < 1.715$, then we have found a feasible solution and we return to the univariate search. Otherwise node I is eliminated and node B will be expanded next.

Two possible cases may occur during the process of branching and bounding: (1) No feasible solution can be found, and all nodes are either cut-off for violating constraints (19A, B, D, E, F and G) or eliminated due to violation of constraint (19C); or, (2) A feasible solution will be found.

In case (1), the current local optimum is a global optimum. In case (2), the univariate search is applied again to obtain a new local optimum and to calculate a new upper bound for each variable, U_p.

Since Z_0 is always decreasing, this new upper bound will be lower than the previous upper bounds. Since the lower bound (L_i) used in our algorithm is an absolute lower bound as required by satisfying the acceptable levels b_1 and b_2, it should remain unchanged for all Z_0. Therefore, the reduction of the upper bound causes the algorithm to converge.

Note that when a new local optimum is obtained, the nodes that have been cut-off due to $T_1(t) > Z_0$ or $E_1 > b_1$, cannot be candidates for feasible solutions because they violate the old Z_0 which implies violation of the new Z_0.

If $T_2(t) < Z_1$, the node t may become feasible for a new Z_1. If it does, the node should be restored to the tree for future expansion. Therefore, we do not have to start an entirely new tree, but rather continue on our tree expansion from those nodes still remaining.

A flowchart in Figure 6 summarizes the procedure for the revised algorithm. It is self-explanatory, we shall not reiterate.

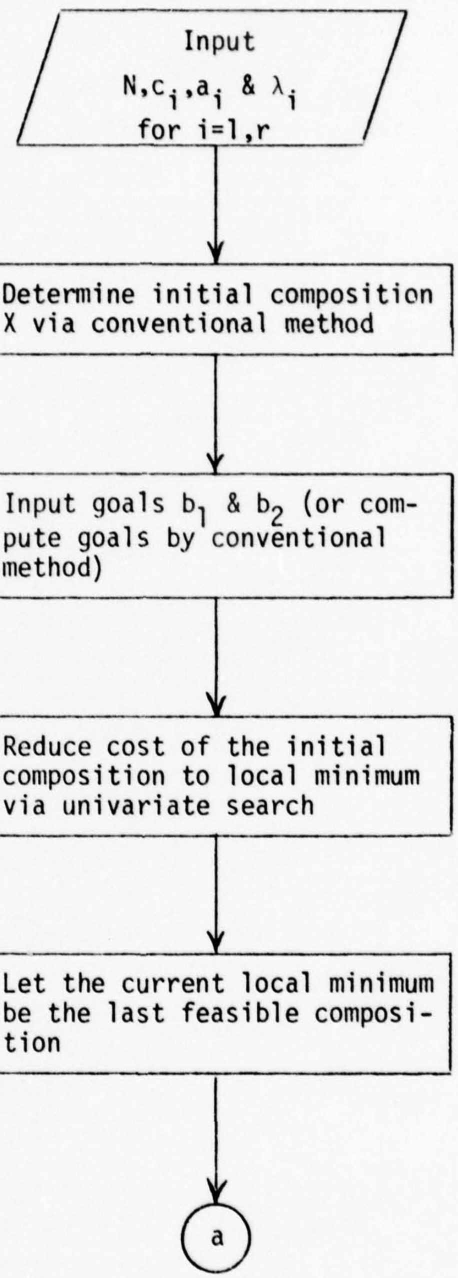


Figure 6. Flowchart for the Revised Algorithm.

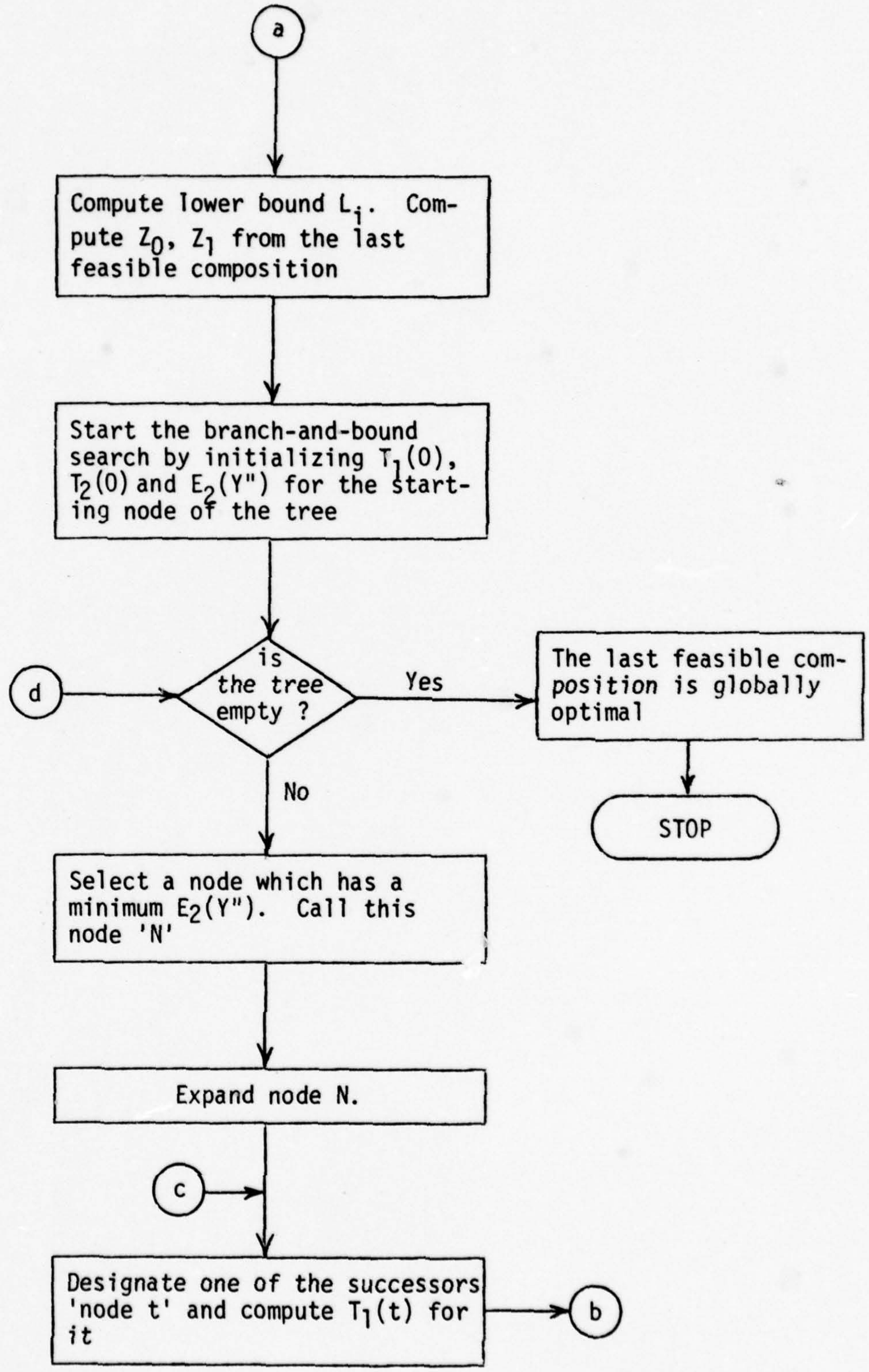


Figure 6. continued

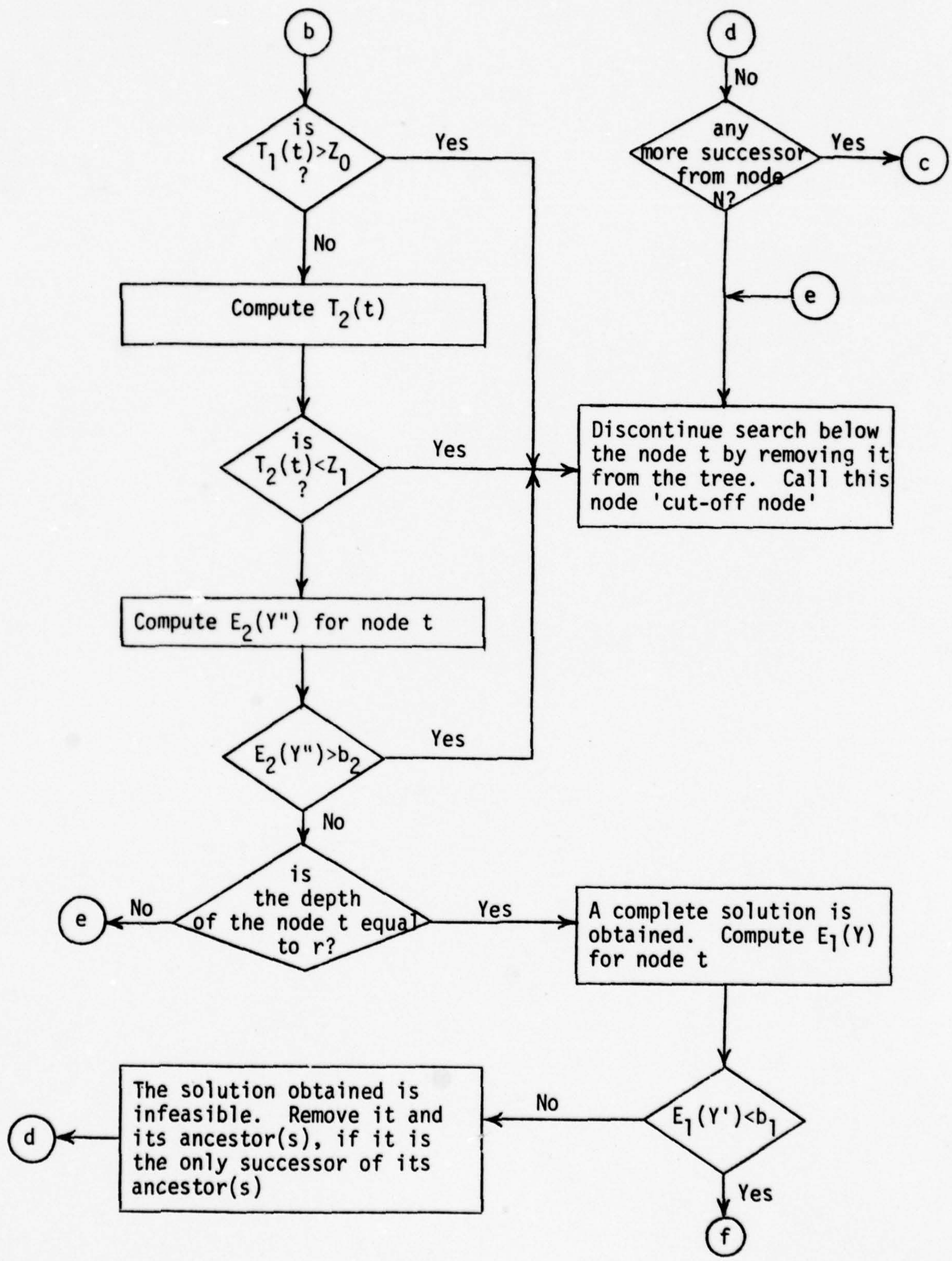


Figure 6. continued

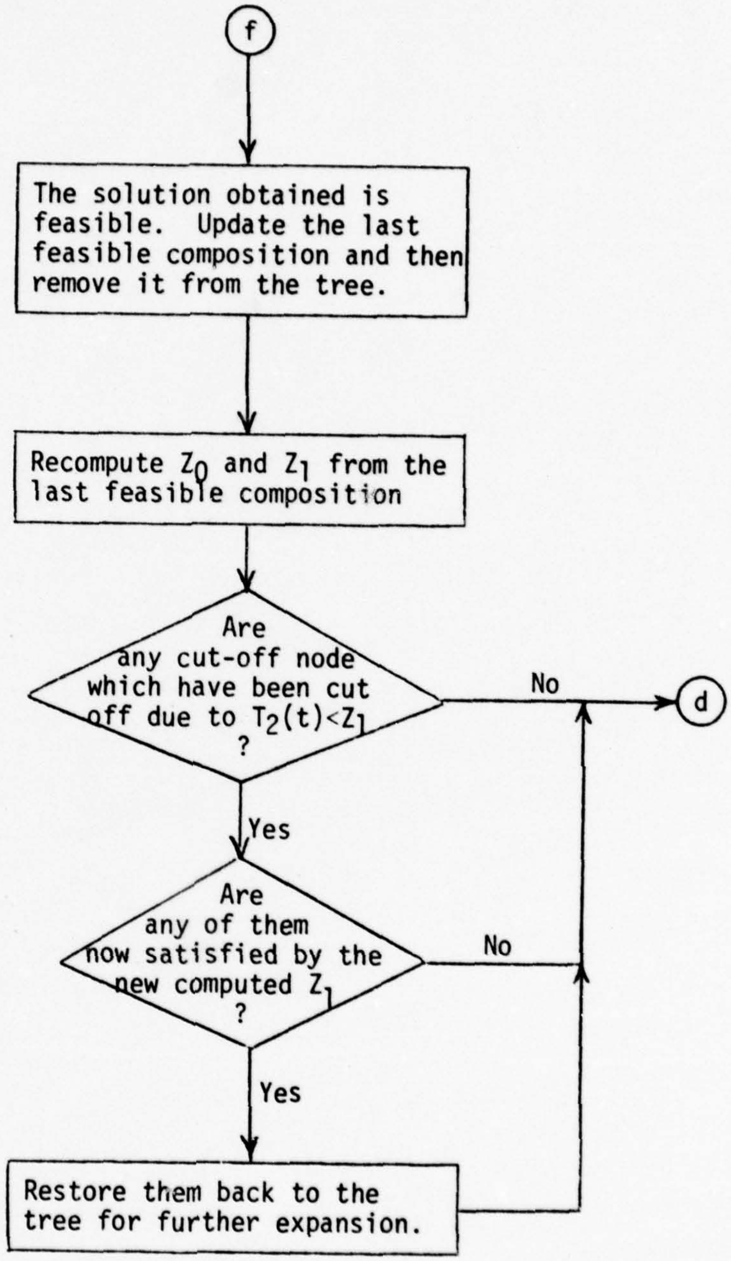


Figure 6. continued

VI. NUMERICAL RESULTS AND CONCLUSIONS

The revised algorithm was programmed in ASCII FORTRAN and tested on a UNIVAC-1110 computer at The University of Alabama. In addition to reporting the computational results, some practical considerations for implementation are recommended. Conclusions concerning this research will follow.

Test Data

The data used in this test was obtained from the first 17 line items of the field data collected at an F-14 base [2]. These data items are summarized in Table 1.

Computational Results

The computational times for the revised algorithm versus the original algorithm are reported in columns 2 and 3 of Table 2. The revised algorithm improves the efficiency in computation. Note that the spares kit is allowed to stock zero units of "essential" parts which have an exceptionally low failure rate.

Practical Considerations for Implementation

Two of the many means of enhancing the solvability of the revised algorithm that we should consider are: the characteristics of the data and the properties of the constraint functions $E_1(X)$ and $E_2(X)$.

Since the number of failures for an item follows a Poisson distribution, theoretically the number of failures can take on values from 0 to ∞ if all possibilities are considered. Practically, however, we may be content with a 95% or 99% reliability for an item.

Table 1

Item Data

Item No. I	No. of Units A(I)	Unit Cost ($) C(I)	Mean Failure Rate per 1800 flying hours $\lambda(I)$
1	1	1206	0.1224
2	1	8116	0.2322
3	1	1041	0.2538
4	1	185	0.0270
5	1	248	0.5508
6	1	165	0.6570
7	1	1141	0.7686
8	1	313	0.1062
9	1	944	0.7038
10	1	1813	0.3636
11	1	2555	2.5758
12	1	363	0.8748
13	1	2416	2.1348
14	1	309	0.3762
15	1	990	0.2556
16	1	6101	0.8334
17	1	493	0.1062

Table 2

Computation Time
(CPU in seconds)

(1) No. of Line Items In the Kit	(2) Original Algorithm	(3) Revised Algorithm (an item may take on 0 unit)	(4) Revised Algorithm (at least 1 unit) Per Item Type
5	0.059	0.094	0.057
6	0.056	0.100	0.081
7	0.138	0.235	0.195
8	0.278	0.396	0.329
9	0.401	0.602	0.379
10	1132.448	1.161	0.647
11	--	2.446	1.055
12	--	3.382	1.160
13	--	5.163	1.210
14	--	7.145	1.261
15	--	10.359	1.316
16	--	25.508	1.701
17	--	77.479	1.755

In the light of this idea, we may obtain a better upper bound, especially for a small random variable.

Finding an upper bound U for X is equivalent to finding the smallest j^* such that

$$\sum_{j=0}^{j^*} q_j \geq 0.95 \tag{20}$$

where q_j is a Poisson density function. Rearranging (20), we obtain a simple expression:

$$\sum_{j=1}^{j^*} \frac{\lambda^j}{j!} \geq 0.95\ e^{-\lambda} - 1 \tag{21}$$

Similarly, the concept of 95% reliability may be utilized to increase the lower bound via equation (14), which in turn can speed up the convergence of the algorithm.

In the case where a minimum of one unit per item is imposed on every line item, the revised algorithm solves the problem with a substantial reduction of time. This can be seen in column (4) of Table 2. The reduction of computation time is due to the low failure rates for most items, which in turn cause the lower bound to be equal to the upper bound. The restriction of at least one unit per item, however, greatly increases the total cost of a kit composition as is shown in Table 3.

By observing the kit composition table of 238 items, we find that many items have approximately the same unit cost, mean failure rate and number of units per item. These items may be grouped and treated as one type. As a result, the revised algorithm can solve a much larger problem size.

Table 3

Comparisons of Total Costs
Revised Algorithm

No. of Items	Lower Bound Determined by Goals b_1 & b_2	Lower Bound Must be at Least One Unit Per Item	Best Local Optimum Prior to Tree Search
5	I*=$7969 F*= 7390	I = 7969 F = 7969	I = 7969 F = 7390
6	I = 8451 F = 7603	I = 8451 F = 8451	I = 8451 F = 7844
7	I = 9108 F = 6160	I = 9108 F = 6811	I = 9108 F = 8097
8	I = 9289 F = 6122	I = 9289 F = 7209	I = 9289 F = 6122
9	I = 9409 F = 6242	I = 9409 F = 7329	I = 9409 F = 6242
10	I = 11461 F = 7980	I = 11461 F = 9059	I = 11461 F = 10212
11	I = 11575 F = 8071	I = 11575 F = 9378	I = 11575 F = 10259
12	I = 11679 F = 8175	I = 11679 F = 9482	I = 11679 F = 10363
13	I = 12669 F = 8169	I = 12669 F = 10472	I = 12669 F = 10467
14	I = 13613 F = 8645	I = 13613 F = 11211	I = 13613 F = 11068
15	I = 14106 F = 8622	I = 14106 F = 11704	I = 14106 F = 10889
16	I = 14596 F = 9112	I = 14596 F = 12353	I = 14596 F = 11379
17	I = 14909 F = 9248	I = 14909 F = 12666	I = 14909 F = 11471

I* = Total cost for the initial kit composition.

F* = Total cost for the final kit composition.

Conclusions

The following conclusions are drawn from this study:

1. Any feasible solution to the constructed optimization model can yield a lower cost investment than does the conventional kit composition while maintaining at least the same levels of support for war readiness.

2. The revised algorithm does improve significantly the computation time for determining an optimum kit composition, though it can only be applied to a limited number of line items.

3. With the imposition of a minimum of one unit per item, the revised algorithm can solve a larger sized problem with much less computation time.

4. Should a global optimum not be mandated, the revised algorithm can solve an even larger problem for a relatively good solution.

5. Grouping of similar items into a single type classification is another strategy to help solve the problem of large size.

VII. REFERENCES

1. Chen, D. S., "Determining A War Readiness Spares Kit," Report of USAF-ASEE Summer Faculty Program, Wright-Patterson AFB, Air Force Logistics Command, 1977.

2. An Analysis of Concepts for War Readiness Spares Kit, Saber Readiness - Delta, Directorate of General Purpose and Airlift Studies, Assistant Chief of Staff, Studies and Analysis, February 1975.

3. Karr, H. W. and M. A. Geisler, "A Fruitful Application of Static Marginal Analysis," Management Science, Vol. 2, 1956, pp. 313-326.

4. Brooks, R. B. S., C. A. Gillen and J. Y. Lu, "Alternative Measures of Supply Performance: Fills, Backorders, Operational Rate, and NORS," RAND RM-6094-PR, August 1969.

5. WRSK/BLSS Authorization Computation System, Internal Report, DAR LOG-MMR-D76-001, U.S. Air Force Logistics Command, January 1976.

6. Dantzig, G. B., "Discrete Variable Extremum Problems," Operations Research, Vol. 5, 1957, pp. 266-277.

7. Everett, H., "Generalized Lagrange Multiplier Method for Solving Problems of Optimal Allocation of Resources," Operations Research, Vol. 11, 1963, pp. 399-417.

8. Feeney, G. J., J. W. Petersen and C. C. Sherbrooke, "An Aggregate Base Stockage Policy for Recoverable Spare Parts," the RAND Corporation, RM-3644-PR, June 1963.

9. Fox, B. L., "Discrete Optimization Via Marginal Analysis," Management Science, Vol. 13, Series A, 1966, pp. 210-216.

10. Fox, B. L., and D. M. Landi, "Optimization Problems With One Constraint," the RAND Corporation, RM-5791-PR, October 1968.

11. Nilson, N. J., "Problem-Solving Methods in Artificial Intelligence," McGraw-Hill, 1971.

```
      COMMON C(25),A(25),LAM(25),K(25),TERM(25),O(25),IUE,IR
      COMMON ENORS,ESHORT,VPGOAL,SPGOAL,PRCONV,TCOST
      COMMON /SEQ/KSEQ(25)
      COMMON /KBOUND/LK(25),KOLD(25),LOLDK(25),TOLD,TLC1
      COMMON /FEACK/PARENT(500),NUNIT(500),FEA
      COMMON /TREE/MULT(25),Z0,Z1,CMIN
      COMMON /SETX/X(1000,25),COT(1000),NGH,NBH,TP(1000),TOPS
     1 ,CCOST(25),SUBT
     COMMON /POISON/QKT(50,25)
      DIMENSION NCOST(500),LEVEL(500),NE(500),STACK(500),ESH(500)
     1 ,TER(100)
      INTEGER A,TCOST,FEA,Z0,Z1,C,TOPS
      INTEGER TP,SUBT,SON
       INTEGER TOP,PARENT,TOPE,X,OK
      REAL LAM
      TOPS=0
       NGH=0
      NBH=1
      DO 5 J=1,999
5      TP(J)=J+1
      TP(1000)=0
      READ*, IUE,IR
      WRITE(6,6)IUE ,IR
 6    FORMAT(1H1,20X,'INPUT DATA :',/26X,'NUMBER OF AIRCRAFT IN A ',
     1 'SQUADRON = ',I4,/26X,'NUMBER OF ITEM TYPES IN THE KIT  =',I5)
      WRITE(6,8)
  8   FORMAT(/1H0,20X,'ITEM','/UNIT COST','/MEAN FAILURE ',
     * 'RATE PER UNIT/',
     1 'NO OF UNITS PER ITEM',/22X,'I',4X,'C(I)',14X,
     2 'LAM(I)',18X,'A(I)')
      DO 10 I=1,IR
      READ*, C(I),LAM(I),A(I)
   10 WRITE(6,9)I,C(I),LAM(I),A(I)
  9   FORMAT(18X,I5,I8,7X,F13.6,12X,I8)
        CMIN=9999999.
        DO 15 I=1,IR
        IF(C(I) .LT. CMIN)CMIN=C(I)
15       CONTINUE
      DO 7 I=1,IR
 7    KSEQ(I)=I
      CALL CUMUO
        CALL CONVE
        CALL BOUNDA
        CALL LOCAL
        CALL MULKI
       CALL SORTLM
       CALL COSTR
      CALL COMBIN
      CALL MULK2
C      DO 20 I=1,IR
C   20 PRINT*, C(I),LAM(I),MULT(I),K(I),LK(I)
C INPUT DATA FOR EXPAN
       DO 30 I=1,IR
30      LOLDK(I)=K(I)
       WRITE(6,300)
  300 FORMAT(//1H0,20X,'IMPROVED LOCAL OPTIMUM KIT COMPOSITION',
```

```
      1 ' VIA BRANCH-AND-BOUND :')
        WRITE(6,305)
  305   FORMAT(36X,'ITEM TYPE, I', 8X,'NUMBER OF UNITS, K(I)')
        WRITE(6,310)(KSEQ(I),K(I),I=1,IR)
  310   FORMAT(40X,I5,20X,I5)
        WRITE(6,315) TCOST
  315   FORMAT(1H0,20X,'TOTAL COST REQUIRED',12X,'= ',I10)
        WRITE(6,320)ENORS, ESHORT
  320   FORMAT(21X,'EXPECTED NO. OF NORS AIRCRAFT  =',F13.6,
      1 /21X,'TOTAL EXPECTED NO OF SHORTAGES =',F13.6)
          TOLD=TCOST
        ENOLD=ENORS
        ESOLD=ESHORT
C *** COMPUTE CUMULATIVE COST
        SUM=0.
        DO 32 I=IR,1,-1
        SUM=SUM+C(I)*MULT(I)
   32   CCOST(I)=SUM
C INITIAL CONDITION OF THE EXPANED TREE
        DO 25 I=1,IR
   25   K(I)=MULT(I)+LK(I)
        CALL SHORT
        ESH(1)=ESHORT
         TOPE=0
        NCOST(1)=0
        NUNIT(1)=0
        PARENT(1)=0
        TOP=0
        NODE=1
           NLAST=2
        LAST=2
         LEVEL(1)=0
        KI=0
C TREE EXPANSION
     95 KI=KI+1
         IF (KI.GT.IR) GO TO 1000
         IF(KI.LT.IR) GOTO 94
          L1=0
          IF(Z1.LT.NCOST(NODE))GO TO 97
         L1=(Z1-NCOST(NODE))/C(KI)
          IF(L1*C(KI) .LT. Z1-NCOST(NODE))L1=L1+1
   97      L2=(Z0-NCOST(NODE))/C(KI)
          IF(L1.GT.L2)GO TO 501
         IF(L2.GT.MULT(KI))L2=MULT(KI)
         L1=L2
         COST=NCOST(NODE)+C(KI)*L1
         GO TO 96
      94 L1=0
         COST=NCOST(NODE)
             DO 91 J5=IR,1,-1
             IF(KOLD(J5) .GE.KI) GOTO 92
   91        CONTINUE
   92        IM=KOLD(J5)
          IF(COST+C(IM).GT.Z0)GO TO 500
      93 L2=(Z0-COST)/C(KI)
         IF(L2.GT.MULT(KI)) L2=MULT(KI)
   96    N9=MULT(KI)-1+LK(KI)
```

```
          L3=L2+LK(KI)
          IF(L2 .EQ. MULT(KI)) GO TO 51
          SUMSD=0.
          DO 50 J4=L3,ND
          CALL DDHORT(KI,J4,DSHORT)
   50     SUMSD=SUMSD+DSHORT
          J4=L2+1
          TER(J4)=SUMSD
   52     IF(J4-1 .LE. L1) GO TO 53
          J4=J4-1
          L3=L3-1
          CALL DDHORT(KI,L3,DSHORT)
          SUMSD=SUMSD+DSHORT
          TER(J4)=SUMSD
          GO TO 52
   51     TER(L2+1)=0.
                SUMSD=0.
          J4=L2+1
          GO TO 52
   53     SON=0
           DO 100 L=L1,L2
           IF(ESH(NODE)+TER(L+1) .GT. SDGOAL) GO TO 206
          IF(KI .EQ. IR) GO TO 101
          IF(COST+CCOST(KI+1) .GE. Z1) GO TO 101
           IF(NBH .EQ.0)PRINT*,'DIMENSION OF X EXCEEDED'
           TOPS=NBH
           NBH=TP(NBH)
           TP(TOPS)=NGH
           NGH=TOPS
C STORE INFORMATION IN TOPS
            COT(TOPS)=NCOST(NODE)
            ICOL=0
            NODEN=NODE
            DO 550 I=1,KI-1
            ICOL=KI-I
            X(TOPS,ICOL)=NUNIT(NODEN)
            NODEN=PARENT(NODEN)
            IF(NODEN .EQ.0) PRINT*,'ERROR IN STORE NODE500'
  550       CONTINUE
            GO TO 206
  101     TOP=TOP+1
          SON = SON+1
          STACK(TOP)=LAST
          NCOST(LAST)=COST
          NUNIT(LAST)=L
          PARENT(LAST)=NODE
          LEVEL(LAST)=KI
          ESH(LAST)=ESH(NODE)+TER(L+1)
          IF(TOPE.EQ.0) GO TO 205
          LAST=NE(TOPE)
          TOPE=TOPE-1
          GO TO 206
  205 NLAST=NLAST+1
              LAST=NLAST
 206          COST=COST+C(KI)
  100 CONTINUE
          IF(SON .EQ. 0) GO TO 1100
```

```
      GO TO 700
501     CONTINUE
C STORE NODE500 HERE
C GET AVAILABLE SPACE FOR NODE500
      ND=MULT(IR)-1+LK(IR)
      L3=L2+LK(IR)
      SUMSD=0.
      DO 523 J4=L3,ND
      CALL DDHORT(IR,J4,DSHORT)
523   SUMSD=SUMSD+DSHORT
      IF(ESH(NODE)+SUMSD .GT. SDGOAL) GO TO 1100
      KKI=IR
      K(IR)= L2
      GO TO 1000
500   KKI=KI
      DO 600 J8=KKI,IR
600   K(J8)=0
      CALL FEASI(KKI,NODE)
      IF(FEA.EQ.1) GOTO 605
      GO TO 1100
605   CALL CHECK(OK)
      IF(OK.EQ.1) GOTO 1006
       GO TO 1100
700   IF(TOP.EQ.0) GO TO 3000
      NODE=STACK(TOP)
      TOP=TOP-1
      GO TO 95
1000  KKI=KI
       CALL FEASI(KKI,NODE)
      IF(FEA.EQ.1) GO TO 1005
       GO TO 1100
1005  CALL CHECK (OK)
      IF(OK.EQ.1) GO TO 1006
       GO TO 1100
1006   TCOST=COST-C(IR)+TLC1
1106   CALL SUBTRT
      CALL COMBIN
      CALL MULK2
       IF(TOLD.EQ.TCOST) GO TO 1100
      DO 39 I=1,IR
39    LOLDK(I)=K(I)
       TOLD=TCOST
      WRITE(6,300)
      WRITE(6,305)
      WRITE(6,310)(KSEQ(I),K(I),I=1,IR)
      WRITE(6,315) TCOST
      WRITE(6,320)ENORS, ESHORT
      ENOLD=ENORS
      ESOLD=ESHORT
       IF(NGH .EQ. 0)GO TO 1100
       CALL GET
      IF(SUBT .NE. 0) GO TO 1300
       IF(FEA.EQ.1) GO TO 1106
1100  TOPE=TOPE+1
      NE(TOPE)=NODE
      IF(TOP.EQ.0) GO TO 3000
      KI=LEVEL(NODE)
```

```
      NP=PARENT(NODE)
C POP
      NODE=STACK(TOP)
      TOP=TOP-1
      IF(LEVEL(NODE).EQ.KI) GO TO 95
 1200 TOPE=TOPE+1
      NE(TOPE)=NP
        KI=LEVEL(NP)
        NP=PARENT(NP)
        IF(NP.EQ.0) GOTO 3000
        IF(LEVEL(NODE).NE.KI) GOTO 1200
      GO TO 95
C ***
C *** RESET THE NODE 500 BACK TO THE TREE
 1300 NN=X(SUBT,IR)
      DO 1305 JS=TOP,2,-1
 1305 STACK(JS+NN)=STACK(JS)
      TOP=TOP+NN
      I=1
      NODEN=1
      DO 1308 JS=2,NN+1
      IF(TOPE .EQ. 0) GO TO 1325
      LAST=NE(TOPE)
      TOPE=TOPE-1
      GO TO 1326
 1325 NLAST=NLAST+1
      LAST=NLAST
 1326 STACK(JS)=LAST
      NCOST(LAST)=X(TOP1,I)*C(I)
      NUNIT(LAST)=X(TOP1,I)
      PARENT(LAST)=NODEN
      LEVEL(LAST)=I
      ESH(LAST)=0
      NODEN=LAST
      I=I+1
 1308 CONTINUE
      DO 1320 I=1,NN
 1320 K(I)=X(TOP1,I)+LK(I)
      DO 1321 I=NN+1,IR
 1321 K(I)=MULT(I)+LK(I)
      CALL SHORT
      ESH(LAST)=ESHORT
      GO TO 1100
 3000 WRITE(6,350)
  350 FORMAT(//1H0,20X,'FINAL GLOBAL OPTIMUM SOLUTION :')
      WRITE(6,305)
      WRITE(6,310)(KSEQ(I),LOLDK(I),I=1,IR)
      WRITE(6,355) TOLD
  355 FORMAT(1H0,20X,'TOTAL COST REQUIRED',12X,'= ',F10.2)
      WRITE(6,320)ENOLD, ESOLD
      STOP
      END
C
C
C**********SUBROUTINE**********
C
C
```

```
      SUBROUTINE FEASI (KKI,NODE)
       COMMON C(25),A(25),LAM(25),K(25),TERM(25),Q(25),IUE,IR
       COMMON ENORS,ESHORT,VPGOAL,SDGOAL,PRCONV,TCOST
      COMMON /FEACK/PARENT(500),NUNIT(500),FEA
      COMMON /KROUND/LK(25),KOLD(25),LOLDK(25),TOLD,TLC1
       INTEGER A,TCOST,C,PARENT
       REAL LAM
      INTEGER FEA
      KKI=KKI-1
      LASTP=NODE
  300 K(KKI)=NUNIT(LASTP)
      IF(KKI.EQ.1) GO TO 400
      LASTP=PARENT(LASTP)
      KKI=KKI-1
      GO TO 300
      ENTRY FEAS
  400 DO 305 I=1,IR
305   K(I)=K(I)+LK(I)
      CALL NORS
      IF(ENORS.GT.VPGOAL) GO TO 310
      CALL SHORT
      IF(ESHORT.GT.SDGOAL) GO TO 310
      FEA=1
      RETURN
  310 FEA=0
      RETURN
       END
      SUBROUTINE CHECK (OK)
       COMMON C(25),A(25),LAM(25),K(25),TERM(25),Q(25),IUE,IR
       COMMON ENORS,ESHORT,VPGOAL,SDGOAL,PRCONV,TCOST
      COMMON /KROUND/LK(25),KOLD(25),LOLDK(25),TOLD,TLC1
      INTEGER OK
      DO 100 I=1,IR
      IF(LOLDK(I).EQ.K(I)) GO TO 100
      OK=1
      GO TO 105
  100 CONTINUE
      OK=0
  105 RETURN
      END
      SUBROUTINE MULKI
      COMMON C(25),A(25),LAM(25),K(25),TERM(25),Q(25),IUE,IR
      COMMON ENORS,ESHORT,VPGOAL,SDGOAL,PRCONV,TCOST
      COMMON /KROUND/LK(25),KOLD(25),LOLDK(25),TOLD,TLC1
      INTEGER POINT,A,TCOST,C,Z0,Z1
      COMMON /TREE/MULT(25),Z0,Z1,CMIN
      REAL LAM
      TCOST=0
      DO 500 I=1,IR
      TCOST=TCOST+K(I)*C(I)
  500 CONTINUE
      Z0=TCOST-TLC1
      Z1=Z0-CMIN
      DO 550 I=1,IR
      T=(1.-0.05)*EXP(LAM(I))
      MULT(I)=Z0/C(I)
      IF(LAM(I) .GE. 5 ) GO TO 600
```

```
          J=0
          TJ=1.
          SUM=0.
      650 SUM=SUM+TJ
          IF(SUM .GE. T) GO TO 700
          J=J+1
          TJ=TJ*LAM(I)/FLOAT(J)
          GO TO 650
      700 IU=J-LK(I)
          GO TO 750
      600 IU=LAM(I)+2*SQRT(LAM(I))-LK(I)
      750 IF (MULT(I).GT.IU) MULT(I)=IU
      550 CONTINUE
          RETURN
          END
           SUBROUTINE SORTLM
          COMMON C(25),A(25),LAM(25),K(25),TERM(25),Q(25),IUE,IR
          COMMON ENORS,ESHORT,VPGOAL,SDGOAL,PRCONV,TCOST
          COMMON /SEQ/KSEQ(25)
            COMMON /KBOUND/LK(25),KOLD(25),LOLDK(25),TOLD,TLC1
           COMMON /TREE/MULT(25),Z0,Z1,CMIN
          INTEGER A,TCOST,C,Z0,Z1
          REAL LAM
          DO 410 I=1,IR
      410 KSEQ(I)=I
      415 KH=1
      420 IF(MULT(KH).LE.MULT(KH+1)) GO TO 430
          T=C(KH)
          C(KH)=C(KH+1)
          C(KH+1)=T
          T=LAM(KH)
          LAM(KH)=LAM(KH+1)
          LAM(KH+1)=T
          IT=A(KH)
          A(KH)=A(KH+1)
          A(KH+1)=IT
           IT=KSEQ(KH)
          KSEQ(KH)=KSEQ(KH+1)
          KSEQ(KH+1)=IT
            IT=MULT(KH)
            MULT(KH)=MULT(KH+1)
            MULT(KH+1)=IT
            IT=K(KH)
            K(KH)=K(KH+1)
            K(KH+1)=IT
            IT=LK(KH)
            LK(KH)=LK(KH+1)
            LK(KH+1)=IT
          GO TO 415
      430 KH=KH+1
          IF(KH.LT.IR) GO TO 420
          RETURN
          END
          SUBROUTINE NORS
          COMMON C(25),A(25),LAM(25),K(25),TERM(25),Q(25),IUE,IR
          COMMON ENORS,ESHORT,VPGOAL,SDGOAL,PRCONV,TCOST
          INTEGER A,TCOST,C
```

```
      REAL LAM
C COMPUTE E(NORS)
      T=0
      DO 700 N=1,IUE
      S=1
       DO 750 I= 1,IR
      K2=K(I)+(N-1)*A(I)
      Q(I)=EXP(-LAM(I))
      TERM(I)=Q(I)
      IF(K2.EQ.0) GO TO 750
      DO 730 J=1,K2
      TERM(I)=TERM(I)*LAM(I)/FLOAT(J)
      IF(TERM(I).LE.1.0E-15) GO TO 750
  730 Q(I)=Q(I)+TERM(I)
  750 S=S*Q(I)
  700 T=T+S
      ENORS=FLOAT(IUE)-T
      RETURN
      END
      SUBROUTINE SHORT
      COMMON C(25),A(25),LAM(25),K(25),TERM(25),Q(25),IUE,IR
      COMMON ENORS,ESHORT,VPGOAL,SDGOAL,PRCONV,TCOST
      COMMON /KBOUND/LK(25),KOLD(25),LOLDK(25),TOLD,TLC1
      COMMON /POISON/QKI(50,25)
      COMMON /SEQ/KSEQ(25)
      INTEGER A,TCOST,C,LOLDK
      REAL LAM
C COMPUTE E(SHORT)
      ESHORT=0
      DO 200 I=1,IR
      JMAX=IUE*A(I)+K(I)
      K1=K(I)+1
      IF(JMAX .GE. 50 ) JMAX=49
      DIF=QKI(JMAX+1,KSEQ(I))*IUE*A(I)
      DO 150 II=JMAX,K1,-1
  150 DIF=DIF-QKI(II,KSEQ(I))
      TERMI=DIF+(IUE*A(I))*(1.-QKI(JMAX+1,KSEQ(I)))
      ESHORT=ESHORT+TERMI
  200 CONTINUE
      RETURN
      END
      SUBROUTINE CONVE
      COMMON C(25),A(25),LAM(25),K(25),TERM(25),Q(25),IUE,IR
      COMMON ENORS,ESHORT,VPGOAL,SDGOAL,PRCONV,TCOST
      COMMON /KBOUND/LK(25),KOLD(25),LOLDK(25),TOLD,TLC1
      INTEGER A,TCOST,C
      REAL LAM
      DO 11 I=1,IR
      K(I)=LAM(I)+0.5
       IF(K(I).LT.1)K(I)=1
   11 CONTINUE
      CALL NORS
       CALL SHORT
      VPGOAL=ENORS
      SDGOAL=ESHORT
      PRCONV=0
      WRITE(6,15) VPGOAL,SDGOAL
```

```
   15  FORMAT(//1H0,20X,'ACCEPTABLE LEVEL FOR EXPECTED NO OF NORS '
      1 ,'AIRCRAFT    = ',F9.5,/
      2 21X,'ACCEPTABLE LEVEL FOR TOTAL EXPECTED NO OF SHORTAGES'
      3 ,' = ',F9.5)
       DO 12 I=1,IR
 12    PRCONV=PRCONV+C(I)*K(I)
       RETURN
        END
        SUBROUTINE LOCAL
        COMMON C(25),A(25),LAM(25),K(25),TERM(25),Q(25),IUE,IR
        COMMON ENORS,ESHORT,VPGOAL,SDGOAL,PRCONV,TCOST
       COMMON /KBOUND/LK(25),KOLD(25),LOLDK(25),TOLD,TLC1
        INTEGER A,TCOST,C
        REAL LAM
        TM=0.5
        TCOST=0
        DO 501 I=1,IR
        K(I)=IFIX(LAM(I)+0.5)
       IF(K(I) .LT. 1 ) K(I)=1
 501    TCOST=TCOST+K(I)*C(I)
       WRITE(6,510)
  510  FORMAT(//1H0,20X,'STARTING KIT COMPOSITIOM :',/36X,'ITEM TYPE,
      1 ' I ', 7X,'NUMBER OF UNITS, K(I)')
       WRITE(6,515)(I,K(I),I=1,IR)
  515  FORMAT(40X,I5,20X,I5)
       WRITE(6,520) TCOST
  520  FORMAT(1H0,20X,'TOTAL COST REQUIRED',12X,'= ',I10)
        CALL NORS
        CALL SHORT
       WRITE(6,320)ENORS, ESHORT
  320  FORMAT(21X,'EXPECTED NO. OF NORS AIRCRAFT  =',F13.6,
      1 /21X,'TOTAL EXPECTED NO OF SHORTAGES =',F13.6)
        IF((ENORS.LE.VPGOAL).AND.(ESHORT.LE.SDGOAL)) GO TO 280
        CALL ADD
 280    DO 281 I=1,IR
 281    KOLD(I)=I
          CALL SUBTRT
        RETURN
        END
       SUBROUTINE ADD
       COMMON C(25),A(25),LAM(25),K(25),TERM(25),Q(25),IUE,IR
       COMMON ENORS,ESHORT,VPGOAL,SDGOAL,PRCONV,TCOST
       COMMON /KBOUND/LK(25),KOLD(25),LOLDK(25),TOLD,TLC1
       INTEGER A,TCOST,C
       REAL LAM
       ITR=1
       DO 51 I=1,IR
 51     KOLD(I)=K(I)
 70    DELMIN=-1.0E20
           CALL SHORT
       ONORS=ENORS
       OSHORT=ESHORT
       DO 80 II=1,IR
       K(II)=K(II)+1
       CALL NORS
       CALL DDHORT(II,K(II)-1,DSHORT)
       ESHOT=OSHORT-DSHORT
```

```
      DELTA=(50.0*(ONORS-ENORS)+(OSHORT-ESHORT))/C(II)
      IF(DELTA.LE.DELMIN)GO TO 90
       DELMIN=DELTA
      IMIN=II
 90   K(II)=K(II)-1
 80   CONTINUE
      IF(DELMIN.LE.-1.0E19) GO TO 98
      K(IMIN)=K(IMIN)+1
      TCOST=TCOST+C(IMIN)
      CALL NORS
      CALL DDHORT(IMIN,K(IMIN)-1,DSHORT)
C     PRINT *,'IMIN=',IMIN,'K(IMIN)=',K(IMIN),'DSHORT=',DSHORT
      ESHORT=OSHORT-DSHORT
       IF((ENORS.LE.VPGOAL).AND.(ESHORT.LE.SDGOAL)) GO TO 98
      ITR=ITR+1
      DO 71 I=1,IR
 71   KOLD(I)=K(I)
      GO TO 70
C98    PRINT *, 'THE LOCAL OPTIMAL COST = ',TCOST
C       PRINT *, 'THE LOCAL OPTIMUN',(K(I), I=1,IR)
C      PRINT *, 'ENORS' ,ENORS,'ESHORT =',ESHORT
  98  RETURN
      END
       SUBROUTINE SUBTRT
       COMMON C(25),A(25),LAM(25),K(25),TERM(25),Q(25),IUE,IR
       COMMON ENORS,ESHORT,VPGOAL,SDGOAL,PRCONV,TCOST
       COMMON /KBOUND/LK(25),KOLD(25),LOLDK(25),TOLD,TLC1
       INTEGER A,TCOST,C
       REAL LAM
           CALL NORS
           CALL SHORT
       ONORS=ENORS
       OSHORT=ESHORT
       DO 80 IK=1,IR
       II=KOLD(IK)
 195   IF (K(II)-1 .LT. LK(II)) GO TO 80
       K(II)=K(II)-1
           CALL DDHORT(II,K(II),DSHORT)
           ESHORT=OSHORT+DSHORT
       IF((ENORS.LT.VPGOAL).AND.(ESHORT.LT.SDGOAL))GOTO190
       K(II)=K(II)+1
       GOTO80
 190   TCOST=TCOST-C(II)
       ONORS=ENORS
       OSHORT=ESHORT
       GOTO195
 80    CONTINUE
      ENORS=ONORS
      ESHORT=OSHORT
       RETURN
       END
       SUBROUTINE GET
       COMMON C(25),A(25),LAM(25),K(25),TERM(25),Q(25),IUE,IR
       COMMON ENORS,ESHORT,VPGOAL,SDGOAL,PRCONV,TCOST
       COMMON /FEACK/PARENT(500),NUNIT(500),FEA
       COMMON /TREE/MULT(25),Z0,Z1,CMIN
       COMMON /GETX/X(1000,25),COT(1000),NGH,NBH,TP(1000),TOPS
```

```
        1 ,CCOST(25),SUBT
          INTEGER A,TCOST,FFA,Z0,Z1,C,TOPS
           INTEGER TP,SUBT
          INTEGER TOP1,PARENT,PTOP1,X,IRI
          REAL LAM
         SUBT=0
          FFA=0
           PTOP1=0
           TOP1=NGH
   506   KI=X(TOP1,IR)
          IF(COT(TOP1) .GT.Z0) GO TO 527
         IF(COT(TOP1)+CCOST(KI+1) .LT. Z1) GO TO 501
         IF(IR-KI .GT. 2) GO TO 1200
         KI=KI+1
         COST=Z0-COT(TOP1)
         L2=COST/C(KI)
         IF(L2 .GT. MULT(KI)) L2=MULT(KI)
         DO 550 LL=1,L2+1
         L=LL-1
         K(KI)=L
         K(IR)=(COST-L*C(KI))/C(IR)
         DO 560 I=1,KI-1
   560   K(I)=X(TOP1,I)
         CALL FEAS
         IF(FFA .EQ. 1) GO TO 559
   550   CONTINUE
         GO TO 507
   559   TCOST=COT(TOP1)+K(KI)*C(KI)+K(IR)*C(IR)
    543   CALL FEAS
          IF(FFA.EQ.1) GO TO 509
           GO TO 507
   527   DO 537 I=KI+1,IR
   537   K(I)=0
         DO 547 I=1,KI
   547   K(I)=X(TOP1,I)
         GO TO 543
  509     TCOST=COT(TOP1)
  507     IF(PTOP1 .EQ.0)GOTO510
           TP(PTOP1)=TP(TOP1)
           TP(TOP1)=NBH
           NBH=TOP1
         IF(FFA .EQ. 1.0) GO TO 1202
           TOP1=TP(PTOP1)
           GO TO 504
  510      NGH=TP(TOP1)
           TP(TOP1)=NBH
           NBH=TOP1
          IF(FFA .EQ.1)GOTO1202
           TOP1=NGH
  504     IF(TOP1 .EQ.0)GO TO 1202
           GO TO 506
  501      PTOP1=TOP1
           TOP1=TP(TOP1)
           GO TO 504
   1200 SUBT=TOP1
  1202    RETURN
          END
```

```
      SUBROUTINE COSTR
      COMMON C(25),A(25),LAM(25),K(25),TERM(25),Q(25),IUE,IR
      COMMON /KBOUND/LK(25),KOLD(25),LOLDK(25),TOLD,TLC1
      COMMON /SETY/X(1000,25),COT(1000),NGH,NBH,TP(1000),TOPS
     1 ,CCOST(25),SUBT
      INTEGER A,TCOST,FEA,Z0,Z1,C,TOPS
      INTEGER TOP,PARENT,TOPE,X,OK,SUBT
      REAL LAM
      DO100 I=1,IR
100   COT(I)=C(I)
      I=1
105   IF(COT(I).GE.COT(I+1))GOTO120
      IK=I
110   TC=COT(IK)
      COT(IK)=COT(IK+1)
      COT(IK+1)=TC
      IC=KOLD(IK)
      KOLD(IK)=KOLD(IK+1)
      KOLD(IK+1)=IC
      IK=IK-1
      IF(IK .EQ.0)GOTO120
      IF(COT(IK) .GE.COT(IK+1))GOTO120
      GOTO110
120   I=I+1
      IF(I .LT.IR)GOTO105
      RETURN
      END
      SUBROUTINE CUMUQ
      COMMON C(25),A(25),LAM(25),K(25),TERM(25),Q(25),IUE,IR
      COMMON ENORS,ESHORT,VPGOAL,SDGOAL,PRCONV,TCOST
      COMMON /SEQ/KSEQ(25)
      COMMON /KBOUND/LK(25),KOLD(25),LOLDK(25),TOLD,TLC1
      COMMON /TREE/MULT(25),Z0,Z1,CMIN
      INTEGER A,TCOST,C,Z0,Z1
      REAL LAM
      COMMON /POISON/QKI(50,25)
      DO 100 I=1,IR
      QJ=EXP(-LAM(I))
      SUM=QJ
      QKI(1,I)=SUM
      DO 105 JK=2,50
      QJ=QJ*LAM(I)/FLOAT(JK-1)
C     IF(QJ .GE. 0.0000001) GO TO 104
C     JKK=JK
C     GO TO 108
      SUM=SUM+QJ
 105  QKI(JK,I)=SUM
      QKI(50,I)=1.0
      GO TO 100
C108  DO 106 JK1=JKK,50
C106  QKI(JK1,I)=1.0
 100  CONTINUE
      RETURN
      END
      SUBROUTINE DDHORT(KI,KKI,DSHORT)
      COMMON C(25),A(25),LAM(25),K(25),TERM(25),Q(25),IUE,IR
      COMMON ENORS,ESHORT,VPGOAL,SDGOAL,PRCONV,TCOST
```

```
      COMMON /KBOUND/LK(25),KOLD(25),LOLDK(25),TOLD,TLC1
      COMMON /POISON/QKI(50,25)
      COMMON /SEQ/KSEQ(25)
      INTEGER A,TCOST,C,LOLDK
      REAL LAM
      M1=IUE*A(KI)+KKI
      IF(M1 .GE. 50) M1=49
      DSHORT=QKI(M1+1,KSEQ(KI))- QKI(KKI+1,KSEQ(KI))
      RETURN
      END
      SUBROUTINE COMBIN
       COMMON C(25),A(25),LAM(25),K(25),TERM(25),Q(25),IUE,IR
       COMMON ENORS,ESHORT,VPGOAL,SDGOAL,PRCONV,TCOST
       COMMON /KBOUND/LK(25),KOLD(25),LOLDK(25),TOLD,TLC1
      COMMON /SEQ/KSEQ(25)
       INTEGER A,TCOST,C
       REAL LAM
 130   ONORS=ENORS
       OSHORT=ESHORT
      DO 230 J1=1,IR-1
      KN=KOLD(J1)
      IF(K(KN)-1 .LT. LK(KN)) GO TO 230
      K(KN)=K(KN)-1
      J3=IR
      J4=J1+1
      CALL DDHORT(KN,K(KN),DSHORP)
      ESHORP=OSHORT+DSHORP
      DO 240 J2=J4,IR
      KP=KOLD(J3)
      K(KP)=K(KP)+1
      CALL NORS
      IF(ENORS .GT. VPGOAL) GO TO 260
C ***
      CALL DDHORT(KP,K(KP)-1,DSHORT)
      ESHORT=ESHORP-DSHORT
C ***
      IF(ESHORT .GT. SDGOAL) GO TO 260
      TCOST=TCOST - C(KN)+ C(KP)
C     PRINT *, 'NEW PT ', (K(I),I=1,IR)
      GO TO 130
 260  K(KP)=K(KP)-1
 240  J3=J3-1
      K(KN)=K(KN)+1
 230  CONTINUE
      WRITE(6,270)
  270 FORMAT(//1H0,20X,'LOCAL OPTIMUM KIT COMPOSITION',
     1 ' VIA UNIVARIATE SEARCH :')
      WRITE(6,275)
 275  FORMAT(36X,'ITEM TYPE, I', 8X,'NUMBER OF UNITS, K(I)')
      WRITE(6,280)(KSEQ(I),K(I),I=1,IR)
 280  FORMAT(40X,I5,20X,I5)
      WRITE(6,520) TCOST
 520  FORMAT(1H0,20X,'TOTAL COST REQUIRED',12X,'= ',I10)
      WRITE(6,300)ONORS, OSHORT
 300  FORMAT(21X,'EXPECTED NO. OF NORS AIRCRAFT  =',F13.6,
     1 /21X,'TOTAL EXPECTED NO OF SHORTAGES =',F13.6)
      ENORS=ONORS
```

```
      ESHORT=OSHORT
      RETURN
      END
      SUBROUTINE MULK2
      COMMON C(25),A(25),LAM(25),K(25),TERM(25),Q(25),IUE,IR
      COMMON ENORS,ESHORT,VPGOAL,SPGOAL,PRCONV,TCOST
      COMMON /KBOUND/LK(25),KOLD(25),LOLDK(25),TOLD,TLC1
      INTEGER POINT,A,TCOST,C,Z0,Z1
      COMMON /TREE/MULT(25),Z0,Z1,CMIN
      REAL LAM
      TCOST=0
      DO 500 I=1,IR
      TCOST=TCOST+K(I)*C(I)
  500 CONTINUE
      Z0=TCOST-TLC1
      Z1=Z0-CMIN
      DO 550 I=1,IR
      IU=Z0/C(I)
      IF(MULT(I) .GT. IU) MULT(I)=IU
550   CONTINUE
      RETURN
      END
      SUBROUTINE BOUNDA
      COMMON C(25),A(25),LAM(25),K(25),TERM(25),Q(25),IUE,IR
      COMMON ENORS,ESHORT,VPGOAL,SPGOAL,PRCONV,TCOST
      COMMON /KBOUND/LK(25),KOLD(25),LOLDK(25),TOLD,TLC1
      COMMON /SEQ/KSEQ(25)
      COMMON /POISON/QKI(50,25)
      INTEGER A,RH,TCOST,C
      REAL LAM
      DIMENSION QT(20)
      DO 300 I=1,IR
      DO 305 JX=1,50
      IF(QKI(JX,KSEQ(I)) .GE. 0.95) GO TO 307
  305 CONTINUE
  307 LOLDK(I)=JX-1
  300 CONTINUE
      TLC1=0.
      DO 310 I=1,IR
      DO 315 IN=1,IUE
      TIME=1
      DO 320 IX=1,IR
      IF(IX .EQ. I) GO TO 320
      NA=(IN-1)*A(IX)
      IF(NA .LT. LOLDK(IX)) GO TO 321
      P=QKI(NA+1,KSEQ(IX))-QKI(LOLDK(IX)+1,KSEQ(IX))
      GO TO 322
  321 P=0.
  322 TIME=TIME*(QKI(LOLDK(IX)+1,KSEQ(IX))+P)
  320 CONTINUE
  315 QT(IN)=TIME
      S=0
      MID=0
      RH=6.0*LAM(I)
      IF (RH.EQ.0) GO TO 390
      DO 220 J1=1,RH
      MID=J1-1
```

```
      T=0
      DO 260 N=1,IUF
      Q(I)=QKI(1,KSEQ(I))
      K1=MID+(N-1)*A(I)
      IF (K1.EQ.0) GO TO 260
      Q(I)=QKI(K1+1,KSEQ(I))
  260 T=T+Q(I)*QT(N)
      S=IUF-T
      IF(S-VPGOAL)390,390,220
  220 CONTINUE
  390 LK(I)=MID
      TLC1=TLC1+C(I)*LK(I)
  310 CONTINUE
      TCOST=TLC1
      WRITE(6,350)
  350 FORMAT(1H0,20X,'ABSOLUTE LOWER BOUNDS FOR ALL ITEM TYPES :'
     1 36X,'ITEM TYPE, I ',10X,'LOWER BOUND L(I)')
      WRITE(6,355)(I,LK(I),I=1,IP)
  355 FORMAT(40X,I5,20X,I5)
      WRITE(6,360) TLC1
  360 FORMAT(1H0,20X,'TOTAL COST FOR THE LOWER-BOUND KIT COMPOSITION
     1 ,' = ',F13.6)
      RETURN
      END
```

INPUT DATA :

NUMBER OF AIRCRAFT IN A SQUADRON = 4

NUMBER OF ITEM TYPES IN THE KIT = 10

ITEM/UNIT COST/MEAN FAILURE RATE PER UNIT/NO OF UNITS PER ITEM

I	C(I)	LAM(I)	A(I)
1	2757	2.358000	1
2	811	.232200	1
3	601	.833400	1
4	278	.147600	1
5	255	2.575800	1
6	241	2.134800	1
7	219	3.414600	1
8	181	.363600	1
9	120	.122400	1
10	114	17.625600	6

ACCEPTABLE LEVEL FOR EXPECTED NO OF NORS AIRCRAFT = 1.98709

ACCEPTABLE LEVEL FOR TOTAL EXPECTED NO OF SHORTAGES = 4.56766

ABSOLUTE LOWER BOUNDS FOR ALL ITEM TYPES :

ITEM TYPE, I	LOWER BOUND L(I)
1	1
2	0
3	0
4	0
5	1
6	1
7	2
8	0
9	0
10	11

TOTAL COST FOR THE LOWER-BOUND KIT COMPOSITION = 4945.000000

STARTING KIT COMPOSITION :

ITEM TYPE, I	NUMBER OF UNITS, K(I)
1	2
2	1
3	1
4	1
5	3
6	2
7	3
8	1
9	1
10	18

TOTAL COST REQUIRED = 11461

EXPECTED NO. OF NORS AIRCRAFT = 1.987090

TOTAL EXPECTED NO OF SHORTAGES = 4.567656

LOCAL OPTIMUM KIT COMPOSITION VIA UNIVARIATE SEARCH :

ITEM TYPE, I	NUMBER OF UNITS, K(I)
2	0
4	0
9	0
1	2
8	1
3	0
5	3
6	2
7	4
10	21

TOTAL COST REQUIRED = 10212
EXPECTED NO. OF NORS AIRCRAFT = 1.963914
TOTAL EXPECTED NO OF SHORTAGES = 4.210705

IMPROVED LOCAL OPTIMUM KIT COMPOSITION VIA BRANCH-AND-BOUND :

ITEM TYPE, I	NUMBER OF UNITS, K(I)
2	0
4	0
9	0
1	2
8	1
3	0
5	3
6	2
7	4
10	21

TOTAL COST REQUIRED = 10212
EXPECTED NO. OF NORS AIRCRAFT = 1.963914
TOTAL EXPECTED NO OF SHORTAGES = 4.210705

LOCAL OPTIMUM KIT COMPOSITION VIA UNIVARIATE SEARCH :

ITEM TYPE, I	NUMBER OF UNITS, K(I)
2	0
4	1
9	1
1	1
8	1
3	0
5	3
6	3
7	4
10	20

TOTAL COST REQUIRED = 7980
EXPECTED NO. OF NORS AIRCRAFT = 1.943458
TOTAL EXPECTED NO OF SHORTAGES = 4.494062

IMPROVED LOCAL OPTIMUM KIT COMPOSITION VIA BRANCH-AND-BOUND :

ITEM TYPE, I	NUMBER OF UNITS, K(I)
2	0
4	1
9	1
1	1
8	1
3	0
5	3
6	3
7	4
10	20

TOTAL COST REQUIRED = 7980
EXPECTED NO. OF NORS AIRCRAFT = 1.943458
TOTAL EXPECTED NO OF SHORTAGES = 4.494062

FINAL GLOBAL OPTIMUM SOLUTION :

ITEM TYPE, I	NUMBER OF UNITS, K(I)
2	0
4	1
9	1
1	1
8	1
3	0
5	3
6	3
7	4
10	20

TOTAL COST REQUIRED = 7980.00
EXPECTED NO. OF NORS AIRCRAFT = 1.943458
TOTAL EXPECTED NO OF SHORTAGES = 4.494062

THE UNIVERSITY OF ALABAMA
COLLEGE OF ENGINEERING

The College of Engineering of The University of Alabama (Tuscaloosa) has an undergraduate enrollment of more than 1,000 students and a graduate enrollment of 90-100. There are approximately 100 faculty members, a significant number of whom conduct research in addition to teaching.

Research is an integral part of the educational program, and interests parallel academic specialities. It is conducted in the classical engineering programs of aerospace, chemical, civil, electrical, engineering hydrology, engineering mechanics, environmental, industrial, mechanical, metallurgical, and mineral engineering. All of these programs offer the master's degree, and five programs offer the educational specialist and doctor of philosophy degrees.

Other organizations on the University campus that contribute to particular research needs of the College of Engineering are the Charles L. Seebeck Computer Center, Geological Survey of Alabama, Marine Environmental Sciences Consortium, Mineral Resources Institute—State Mine Experiment Station, Natural Resources Center, U.S. Bureau of Mines, Tuscaloosa Metallurgy Research Center, and the Research Grants Committee.

This University community provides opportunities for interdisciplinary work in pursuit of the basic goals of teaching, research, and public service.